DESIGN OF A MULTIPROCESSOR DSP-BASED MACHINE SUITED FOR INTENSIVE REAL-TIME APPLICATIONS

by
Sylvain Martel

A thesis submitted to
the Faculty of Graduate studies and Research
in partial fulfillment of the requirements for the degree of
Master of Engineering

Department of Electrical Engineering
and
Biomedical Engineering Unit
McGill University
Montréal. Québec. Canada
January. 1989

ABSTRACT

A multiprocessor system based on 32 digital signal processors (DSPs) has been designed to meet the demanding computational and control requirements of an apparatus (for studying the mechanical and optical properties of single living muscle cells) containing a very-high performance micro-robot and a 3-D laser scanning microscope. A novel architecture has been developed to resolve some of the drawbacks encountered in loosely and tightly coupled systems, as well as offering a fast I/O interface. Each processor has a separate parallel path to every other processor via dual-port memory. The fully configured system provides 1 Gflops (theoretical maximum) of 32-bit floating-point performance and 5 Gbps I/O capability.

RÉSUMÉ

Un système à multi-processeurs basé sur 32 processeurs pour signaux digitaux (DSPs) fut développé pour satisfaire la demande pour exécuter les calculs et le contrôle requis pour un appareillage (pour étudier les propriétés mécaniques et optiques de cellules d'un muscle vivant) contenant une paire de micro-robots à très haute performance et un microscope à balayage optique en 3 dimensions par rayon laser. Une nouvelle architecture fut développée afin de résoudre quelques lacunes observées dans les systèmes avec couplages relâchés ou serrés, tout en offrant une interface rapide d'entrées et de sorties. Chaque processeur bénéficie d'un accès parallèle aux autres processeurs à travers des modules de mémoire permettant des accès simultanés par les deux côtés du module. Le système entier peut offrir une performance jusqu'à un milliard d'opérations de 32 bits à point flottant par seconde (maximum théorique) et jusqu'à 5 milliards de bits par seconde pour les entrées et sorties.

ACKNOWLEDGEMENTS

This thesis is the result of a design process involving many modifications mainly due to suggestions made by my supervisor Dr. Ian W. Hunter, and good collaborators Dr. Poul M.F. Nielsen and Mr. Serge Lafontaine. Their constructive criticisms and recommendations transfomed a simple machine based on eight independent DSP modules into a fully connected 32 DSP-based machine with very high performance and flexibility.

I particularly thank Dr. Hunter for giving me the opportunity to try new concepts for parallel computers. These innovations permitted me to expand my knowledge as well as increasing the overall performance of the computer.

Dr. Nielsen helped me to visualize better the real problems that would be run on to the computer. His remarks greatly enhanced the design so that it would respond more efficiently to specific applications and environments.

I appreciated the help from Mr. Lafontaine regarding the software and particularly the connection between the system and the MicroVAX.

Special thanks to Tilemachos Doukoglou (Telly), a good friend who contributed to the good working environment in the laboratory.

Finally, I would like to thank the Medical Research Council of Canada and the Natural Science and Engineering Research Council of Canada for funding of the above project. I also thank Texas Instruments for the donation of the TMS320C30 optimizing C compiler, linker, simulator, and a few thousand ICs.

TABLE OF CONTENTS

Abstract (page i)
Résumé (page ii)
Acknowledgements (page iii)
Table of contents (page iv)
List of figures (page vii)
Glossary of abbreviations (page viii)

CHAPTER ONE - **Introduction** (page 1)
- 1.1 Environment (page 1)
- 1.2 System requirements (page 1)
- 1.3 Existing computers (page 3)
- 1.4 Overview (page 4)

CHAPTER TWO - **Basic system** (page 6)
- 2.1 Digital signal processor (pagc 6)
- 2.2 System cost vs throughput (page 7)
- 2.3 Design philosophy (page 9)
- 2.4 Circuit boards (page 10)
- 2.5 DSP board (page 10)

CHAPTER THREE - **Memory** (page 12)
- 3.1 Dynamic RAM interface (page 12)
  - 3.1.1 Refresh timer (page 12)
  - 3.1.2 Memory timing controller (page 14)
  - 3.1.3 Dynamic RAM controller (page 19)
  - 3.1.4 Memory driver (page 20)
  - 3.1.5 DRAM outputs and syndrome latches (page 20)
  - 3.1.6 DRAM access time (page 20)
  - 3.1.7 Error detection and correction (page 21)
- 3.2 Static RAM block (page 23)

TABLE OF CONTENTS

3.2.1 Choice of the bus (page 24)
3.2.2 Access time for the static RAM (page 24)
3.2.3 Static RAM configuration (page 24)
3.2.4 Bank switching (page 25)
3.2.5 Decode section (page 26)
3.2.6 Number of address drivers (page 29)
3.2.7 Number of transceivers (page 29)
3.2.8 Transceivers output enable control (page 31)
3.2.9 Read/write line (page 32)
3.2.10 Total access time (page 33)
3.2.11 Memory re-allocation (page 33)
3.2.12 PAL-decode approach (page 34)

CHAPTER FOUR - **Main components of the DSP board** (page 36)
4.1 Multicast system (page 36)
4.2 Connection bus interface (page 38)
4.3 Wait-state generator (page 39)

CHAPTER FIVE - **I/O subsystem** (page 40)
5.1 I/O subsystem requirements (page 40)
5.2 Serial-to-parallel and parallel-to-serial conversions (page 41)
5.3 Number of serial channels (page 43)
5.4 Selection of the clock (page 43)
5.5 Serial channel controls (page 44)
5.5.1 Serial control registers (page 45)
5.6 Counters (page 46)
5.7 Polling and interrupt (page 46)
5.7.1 Status and interrupt registers (page 47)
5.7.2 Mask register (page 48)
5.8 Serial control block (page 48)
5.9 Interrupt controller (page 49)
5.10 I/O interface (page 50)
5.11 Decoders (page 51)

TABLE OF CONTENTS

CHAPTER SIX - Dual-port RAM modules (page 52)

- 6.1 Dual-port module interface (page 53)
  - 6.1.1 Dual-port data bus interface (page 53)
  - 6.1.2 Receivers (page 53)
  - 6.1.3 Interrupt identification scheme (page 53)
- 6.2 Dual-port RAM (page 54)
- 6.3 Busy lock-up (page 57)

CHAPTER SEVEN - Interface board (page 58)

CHAPTER EIGHT - Conclusions (page 60)

APPENDICES - Schematics

- Appendix A - DSP board
- Appendix B - Local memory board
- Appendix C - Dual-port module
- Appendix D - I/O subsystem
- Appendix E - Interface board
- Appendix F - DRAM interface
- Appendix G - Unit list
- Appendix H - External registers
- Appendix I - Basic memory map

References

LIST OF FIGURES

Figure 1.1 - Architecture of the parallel computer (page 2)
Figure 2.1 - Cost per flops for both architectures (page 9)
Figure 2.2 - Simplified block diagram of the DSP board (page 10)
Figure 3.1 - Block diagram of the DRAM interface (page 13)
Figure 3.2 - The three selectable DRAM access modes (page 16)
Figure 3.3 - Flowchart of the different operations of the timing controller (page 18)
Figure 3.4 - A standard decoding scheme (page 26)
Figure 3.5 - A one-level decoding scheme requiring two chip enables (page 27)
Figure 3.6 - A one-level decoding scheme requiring one chip enable (page 28)
Figure 3.7 - Block diagram of the 512Kx32 bit memory board (page 30)
Figure 3.8 - A PAL-based decoding scheme (page 34)
Figure 4.1 - Simplified block diagram of the multicast system (page 37)
Figure 5.1 - I/O subsystem for the serial channels (page 41)
Figure 5.2 - Basic circuitry of one serial channel (page 42)
Figure 5.3 - Basic configuration for polling and interrupt (page 47)
Figure 5.4 - The serial control block (page 48)
Figure 6.1 - Block diagram of one dual-port module (page 55)
Figure 7.1 - State diagram for the interface board (page 58)
Figure 7.2 - Block diagram of the interface board (page 59)

GLOSSARY OF ABBREVIATIONS

a - Relative cost of memory and dual-port module respectively to the cost of DSPs

A/D - Analog-to-digital

AND - AND logic

ASIC - Application specific integrated circuit

b - Relative cost of memory and dual-port module respectively to the cost of DSPs

bps - Bits per second

CAS - Column address strobe

C_d - Cost of one dual-port module

$C_{i/o}$ - Input/output capacitance

C_m - Cost of memory

C_p - Cost of one digital signal processor

C_s - Cost of the system

D/A - Digital-to-analog

DIP - Dual-in-line package

DMA - Direct memory access

DPM - Dual-port module

DPMI - Dual-port module interface

DRAM - Dynamic RAM

DSP - Digital signal processor

EDAC - Error detection and correction

EIA - Electronic industries association

EOC - End of count

flops - Floating-point operations per second

G - Giga

Hz - Hertz

IC - Integrated circuit

IDT - Integrated Device Technology

I_{ih} - Input high current

I_{il} - Input low current

IIS - Interrupt identification scheme

I_{ix} - Input load current

I/O - Input/output

I_{oh} - High-level output current

GLOSSARY OF ABBREVIATIONS

I_{ol} - Low-level output current

K - Kilo

M - Mega

ms - millisecond

MSB - Most significant bit

MTBF - Mean time between failures

MUX - Multiplexer

N - Number of DSPs in the system

NAND - Negative AND logic

NOR - Negative OR logic

ns - Nanosecond

OR - OR logic

PAL - Programmable array logic

PCB - Printed circuit board

pF - Picofarad

PLCC - Plastic leaded chip carrier

R - Read access time

RAM - Random access memory

RAS - Row address strobe

ROM - Read only memory

R/W - Read/write

SCD - Static column decode

SCR - Serial control register

SLM - Scanning laser microscope

SMT - Surface mounted technology

SOIC - Small outline integrated circuit

SRAM - Static RAM

T - Expected or desirable throughput

t_{acc} - Access time

t'_{acc} - New access time

T_p - Processor throughput

t_{pd} - Propagation delay

t'_{pd} - New propagation delay

GLOSSARY OF ABBREVIATIONS

t_{phl} - Propagation delay time. high-to-low output

t_{pzh} - Output enable time

T_s - System throughput

t_{sd} - Set-up time

uA - Micro-Ampere

V - Volt

Vcc - Supply voltage (+5 volts)

W - Number of wait states or write access time

ws - Wait state

XOR - Exclusive OR logic

/ - Indicate negation as for /AB16 used in PAL equations (AB16 = 0)

* Signals and IC numbers have not been included in the glossary of abbreviations and symbols.

CHAPTER ONE

INTRODUCTION

1.1 Environment

The development of the parallel processor was motivated by the substantial computation and control requirements of our laboratory. The laboratory includes a scanning laser microscope (SLM) which is capable of very high resolution, low noise, and quantitative 3-dimensional optical measurements on single living muscle fibers [1]. The apparatus includes two high performance 3-axis micro-robots which perform mechanical experiments on the fibers during imaging [2]. Each micro-robot has 6 actuators each of which must be controlled with a bandwidth of 100 kHz using nonlinear digital control schemes. The 10^7 dynamic range of the micro-robots' displacement requires high-speed 32-bit floating-point and computation.

1.2 System requirements

The system must be flexible and capable of high performance. Flexibility is attained by software. thus the system must be programmable. High performance at reasonable cost can be achieved with multiple processors. The system must be able to deal with long and complex algorithms in real time. A solution to these requirements is to provide a communication pattern between processors so that multitasking and partitioning are possible. This leads to consideration of parallel supercomputers [3]. Because our applications require high rates of memory access, and since contention problems result in bottlenecks in shared memory systems, a sufficiently large independent memory must be dedicated to each processor in order to minimize processor wait-states due to the contention problem. The same problem arises with interprocessor communication. Our solution involves a novel architecture [4][5] that does not require bus arbitration logic (see Figure 1.1). New dual-port RAMs (random access memory) offer many features necessary in interprocessor communication. It is feasible to design communication modules which include access arbitration logic, sufficient buffer area, as well as interrupt facilities.Such a scheme provides a fast and efficient solution for interprocessor communication and signalling, without the drawback of long communication times associated with loosely coupled systems, or the contention problem associated with tightly coupled systems. The interprocessor communication must be efficient. tne system must allow broadcast and multicast options. Also zero wait-state memory accesses must be performed in order to maintain the high throughput of the processor.

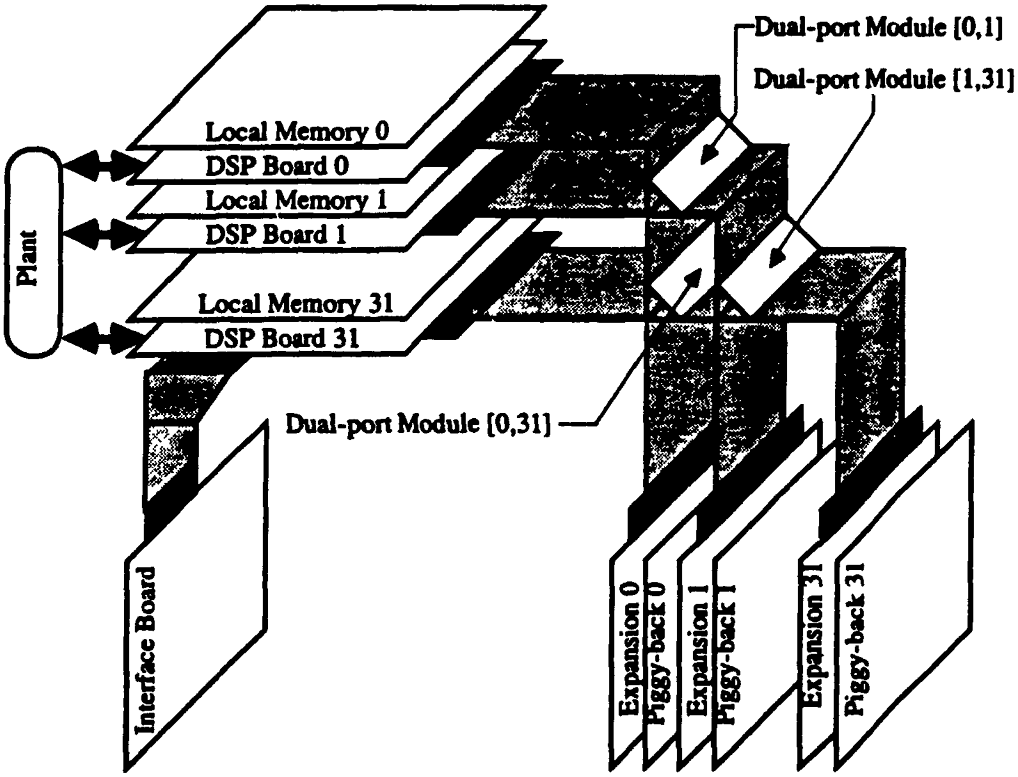


Figure 1.1 - Architecture of the parallel computer.

Being an experimental system, it must be flexible to adapt itself to a changing environment. Finally, the I/O subsystem must provide a large number of high speed input and output channels in order to support open and closed loop control schemes. It must also provide flexible mechanisms to control A/D and D/A converters without compromising the processor's computing performance.

1.3 Existing computers

Several existing computer systems have the required peak performance of 1 Gflops. Reference [35] surveys the main characteristics of these supercomputers. The Cray 2 [36], the Cray 3, the ETA-10 [37], the IBM GF-11 [38], and the Cedar [39] offer, when fully configured, peak performances of 2 Gflops, 16 Gflops, 10 Gflops, 11 Gflops, and 3.2 Gflops respectively. The NCE SX-2, the Hitachi S-810 [40], and the IBM RP3 [43] achieve peak performance of 1.3 Gflops, 840 Mflops, and 800 Mflops respectively.

Looking at the peak performance alone is not sufficient. For our purposes a multiprocessor system is preferable to a uniprocessor. This is mainly due to the fact that a system involving many processors allows more flexibility towards modular expansion and configuration. The initial cost can also be much lower. Much of the above mentioned computers have a low number of processors to reduce the contention problem. Only the IBM RP3 and the Cedar computers can afford many processors through the use of special interconnection networks, distributed memory, and/or hierarchical schemes. Many other multiprocessor systems are available but provide far less performance.

Unfortunately, high-speed processor operations alone are not sufficient for the intensive I/O multiprocessing required by our real-time environment. A configurable interface able to communicate with up to 512 external A/D and D/A converters is of prime importance. Since our system will be almost entirely dedicated to real-time applications, the number of I/O channels must be sufficient to match with the high computational performance of the computer. It seems that current systems offering the required computational facilities, lack I/O throughput performance. Some I/O architectures of new parallel computers indicate a clear trend in demanding higher I/O bandwidth. For example, 40 peripheral devices can be attached to the Cray 2.

Another modem supercomputer, the Convex C-1 [41][42], has five intelligent channel control units able to support up to 80 channels. However, these computers lack the throughput performance (input-computation-output lag) required for real-time control.

Since existing computers with the required peak performance are unable to support our I/O requirements. a new architecture based on 32 high-performance floating-point DSPs. a fast and efficient interconnection network, a distributed memory scheme. and a powerful I/O subsystem has been developed and is presented in this thesis.

1.4 Overview

The following chapters constitute an expanded version of a paper we submitted to IEE Proceedings Part G: Electronic Circuits and Systems. This thesis describes the main steps in the digital design of a parallel supercomputer based on a novel fully connected architecture. Several parts of the design phase were omitted in order to keep the description as clear as possible without getting too technical, although detailed descriptions of the most important design concepts at strategic portions of the computer are given. Detailed schematics and PAL (programmable array logic) equations are provided in the Appendices for reference. Timing diagrams are not included since too many of them would be required. Calculations for the current driving capabilities (fan-in and fan-out), capacitance, and timing characteristics have been carefully checked. The strategic parts of the circuits have been tested experimentally except for the DRAM (dynamic RAM) interface. All PAL equations have been verified by simulation except for the timing controller and the bus/EDAC (error detection and correction) controller. The multiplexers described in Section 5.4 have not been included in the schematics in order to simplify the circuit and to avoid the possible drawbacks described in Section 5.5.1. This involved a slight modification in the synchronization circuits of the I/O channels.

The bus connecting the I/O subsystem, the external registers, and the interface board to the processor, is referred to as the I/O or secondary bus. The bus joining the connection bus and the local SRAM (static RAM) memory to the processor, is refered to as the primary or memory bus. The bus connecting the DSP (digital signal processor) board to the dual-port modules and the expansion board, is referred to as the connection or communication bus. The dual-port module is referred to as the connection module in reference [5].

The main innovations of this computer are: (by order of importance)

(1) The interprocessor communication scheme
(2) The multicast system
(3) The multiplexed interrupt identification scheme
(4) The reconfigurable I/O subsystem
(5) The serial control block
(6) The memory re-allocation feature
(7) The initialization program preload circuitry
(8) The automatic switch for DRAM access

Chapter 2 describes briefly the choice of the processor. A short analysis of the system cost versus the throughput is also included. This chapter ends with a short description of the design philosophy, the different boards, as well as an initial description of the DSP board.

Chapter 3 is divided into two parts. The first section deals with the dynamic RAM interface which is not part of the basic computer system. A relatively detailed description of this part of the design was mandatory to determine the signals required by the connection bus for interfacing a memory expansion module in the future. The second part deals with a zero wait-state local memory module implemented with static RAMs. Emphasis is given to the memory interface which permits zero wait-state accesses in all cases.

Chapter 4 describes the main parts of the DSP board. Chapter 5 deals with the I/O subsystem. The dual-port modules are described in some detail in Chapter 6. It consists mainly of the dual-port module interface and the dual-port RAM itself. Consideration is also given to access time requirements.

Chapter 7 describes the board which links a MicroVAX to the system. The interface board itself only performs the handshakes with the MicroVAX for exchange of information, and provides a temporary buffer area. A DSP is assigned as a communication processor and brings some intelligence to the interface.

CHAPTER TWO

BASIC SYSTEM

2.1 Digital signal processor

The choice of the processor is an important aspect to consider since it represents the heart of the system. It must be flexible enough to incorporate improvements. Many DSP techniques are still developing. and therefore their algorithms tend to change. This implies that DSP systems need to be programmable to accomodate revised algorithms. It must support 32-bit floating-point arithmetic and provide high computing power. It must support direct memory access (DMA) on the memory bus. The command library should be complete, and external interrupt facilities must be provided. Ideally. the processor should have separate address lines in order to avoid the complexity of multiplexed buses and the delay involved in such schemes. Also, it should be able to address a sufficiently large amount of data and instructions. Finally. an I/O bus should be provided to reduce the memory bus traffic when I/O channels are accessed. A high level language with appropriate tools such as an optimizing compiler. linker, and simulator must be available on an appropriate operating system to lower the system software development costs.

Many alternatives are offered to the designer. The custom IC (integrated circuit) approach yields the fastest throughput but at the expense of flexibility. system fault tolerance. expandability, cost and design time. Array processors have long been the accepted solution for the research laboratory. However. as integrated circuit technology has matured. digital signal processing has migrated from the array processor to the bit-slice processor to the single-chip processor.

Texas Instruments (TI) TMS320C30 is an example of a single-chip digital signal processor [6] that meets our requirements by providing 32-bit 33 Mflops (million floating-point operations per second) performance. a 16 Mword address space. four external interrupts. and two independent parallel buses. It is also programmable in the high-level language C. - a factor that lowers system software costs and shortens design time. Unlike conventional von Newmann architectures. the Harvard architecture of the TMS320C30 uses a dual-bus design for parallel fetching of code and data to support high-speed complex arithmetic. It is also one of the best DSP chips in the market today according to [7].

. We had initially decided to use AT&T's DSP32 (8 Mflops) in our design. Indeed we thank AT&T for a generous donation of 8 DSP32's together with a development system and extensive software. However the DSP32 had numerous deficiencies and was therefore abandoned in favour of the TMS320C30. The design of our parallel computer was facilitated considerably by a non-disclosure agreement with TI which resulted in us receiving detailed timing data as they became available.

2.2 System cost vs throughput

The cost of the system must be taken into account prior to the design phase. The throughput expected and the price of the main components should be considered before choosing the structure of the parallel computer. Let us define:

C_s : cost of the system
C_p : cost of one digital signal processor
C_m : cost of memory
C_d : cost of one dual-port module
N : number of DSPs in the system

For a typical tightly coupled parallel computer:

$$C_s = C_m + NC_p .$$

In terms of processor cost:

$C_m = aC_p$ and $C_d = bC_p$, a and b being the relative cost of memory and dual-port module respectively to the cost of a processor.

Then $C_s = (a + N)C_p$.

For the present system:

$$C_s = NC_m + NC_p + [[N(N - 1)]/2]C_d$$

$C_s = NC_p[(a + 1) + [(N - 1)/2)]b]$.

It should be noted that the price of the proposed system increases faster than for a shared bus and memory computer mainly due to the cost of dual-port memories. Now, if we define:

T_s : system throughput
T_p : processor throughput
T : expected or desirable throughput,

then the throughput of the present system can be defined as:

$T_s = NT_p$.

For a typical system, with a bus utilization or a main memory access rate of 0.25, C_s can be determined by $(a + N)C_p$ for $0 < T < 5T_p$. A sophisticated cache scheme can be implemented to reduce the bus utilization, thus increasing N, but maximum performance is only obtained with perfect synchronization between all processors in accessing the bus, which is a complex and almost impossible task to perform. Also, the cache will not avoid all bus accesses. Past experience has demonstrated that a relatively small number of processors can be connected to the same bus in order to obtain reasonable performance.

The proposed architecture does not face this problem. The total number of processors has been restricted to 32 for reasons of cost and design parameters. With 32 DSPs, the partition of C_s is as follows:

Cost of processors = $NC_p = 32\ C_p$
Cost of local private memories = $NC_m = a * 32\ C_p$
Cost of dual-port memories = $[N(N - 1)/2]C_d = b * 496\ C_p$.

Figure 2.1 shows the relation between the cost per flops for the two architectures.

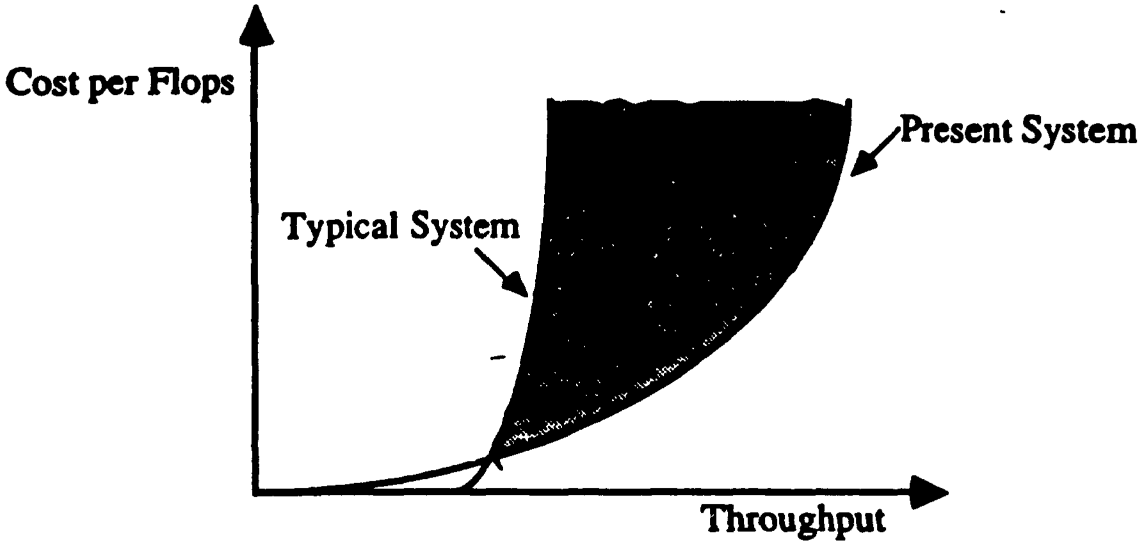


Figure 2.1 - Cost per flops for both architectures.

2.3 Design philosophy

In designing the parallel supercomputer, we always kept in mind that speed should be maximized. Special techniques and devices with short propagation delays have been chosen. Although we have tried to simplify the circuits as much as possible, the resulting PCBs are still complicated. To reduce this complexity, extensive use of PALs has been adopted. There are four reasons for this: Firstly, the input/output delay is constant, even for very complex combinatorial functions, since the internal structure of PALs is always two-level. Any change within the same device does not affect system timing. Secondly, at the prototype level, possible combinatorial design faults can be often dealt with without modifications to the PCB, provided the missed or unwanted signals are already used in the same PAL. Thirdly, with PALs the structure of the PCB becomes very regular. And fourthly, the use of PALs greatly reduces the number of ICs, thus reducing the PCB layout complexity. We also selected ICs that are available in surface mount technology (SMT) well appreciated for their small package.

Finally, the design was made flexible to adapt to different environments. Cost and power consumption were also taken into account.

2.4 Circuit boards

In order to build the basic system, five different types of boards had to be designed: The DSP board constitutes the computing power of the system and was also the most complex board to design: The local memory board was straighforward to design since it consists of regular arrays of high-density static RAMs; The I/O interface board was relatively complex since it had to be flexible enough to interface to a wide variety of A/D's. D/A's, and other I/O devices. The dual-port module is the smallest board in the system and was relatively easy to design: And finally, the interface board which provides the means of communication between the MicroVAX and the system required a moderately complicated design.

If the whole system is populated (excluding any expansion boards), 32 DSP boards, 32 local memory boards, 32 I/O subsystem boards, 496 dual-port modules, and one interface board would be required.

For possible future expansion of the system memory a sixth board, called the expansion board, containing DRAMs with an EDAC has been designed.

2.5 DSP board

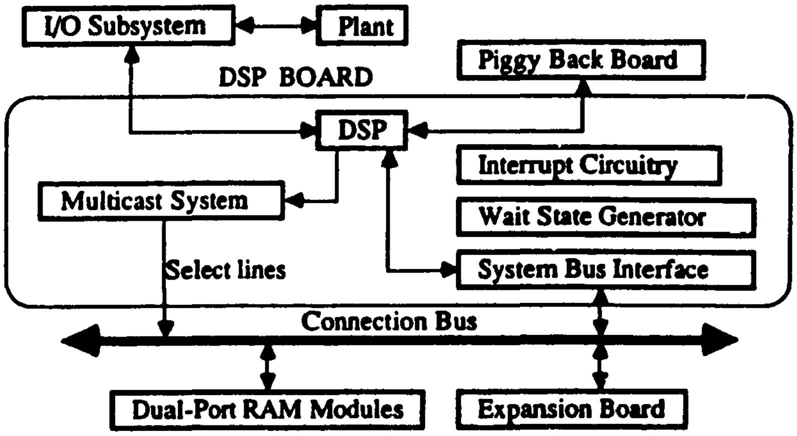


Figure 2.2 - Simplified block diagram of the DSP board.

CHAPTER TWO - BASIC SYSTEM

Surrounding the TMS320C30 is the circuitry required to support the processor. The DSP board provides processing capability, the multicast system, external interrupt circuitry, decoders and external registers, and connection bus interface. The simplified block diagram is shown in Figure 2.2.

CHAPTER THREE

MEMORY

Two 1Kx32 bit blocks of single cycle access internal RAM are available to the programmer. To expand the size of zero wait-state memory, a SRAM block can be added on the piggy-back board adjacent to the DSP board. Up to 512Kx32 bit of additional SRAMs can be implemented on this adjacent board. For larger amount of local memory, DRAMs must be considered. In this case wait-states will exist, despite the fact that special access modes are implemented to minimize the access time. DRAMs will normally be implemented on the expansion board. This board would contain all the DRAM interface and up to 16 Mwords of DRAMs divided in a section of 4 Mwords and another section of 12 Mwords. Since the expansion board can be used to extend the local memory or to add a co-processor, both alternatives must be considered in designing the system and to provide an appropriate connection bus interface. The next sections describe the DRAM interface which will be used as an indicator of the required signals that must propagate through the connection bus.

3.1 Dynamic RAM interface

A refresh timer, memory timing controller, DRAM controller, memory driver, static column detector, DRAM output latches, syndrome latch, and a fault tolerant system implemented with an EDAC (Error Detection And Correction) constitute the DRAM interface [8] as shown in Figure 3.1.

3.1.1 Refresh timer

The refresh timer is used to signal to the timing controller that it should execute a refresh cycle as required by the dynamic memory. Design and operation of the DRAM allow only one row to be refreshed at a time. Typical 1Mx1 bit DRAM has 512 rows that must be refreshed every 8 ms. The array is actually 1024 rows by 1024 columns, but it operates electrically like two half arrays of 512 rows by 1024 columns. During refresh, every row is treated as if it runs through both halves of the array, refreshing 2048 column locations (bit cells) per row. This design results in fewer refresh cycles required to recharge the entire array.

Many alternatives exist to perform this task. Refresh can be performed in either a single burst of 512 consecutive refresh cycles, or distributed over time, one refresh cycle every 15.6 microseconds on average. Since the TMS320C30 is not able to reach the DRAM during refresh cycles, we opt for the distributed option since we must avoid requiring the DSP to wait too many cycles in time critical sections of code.

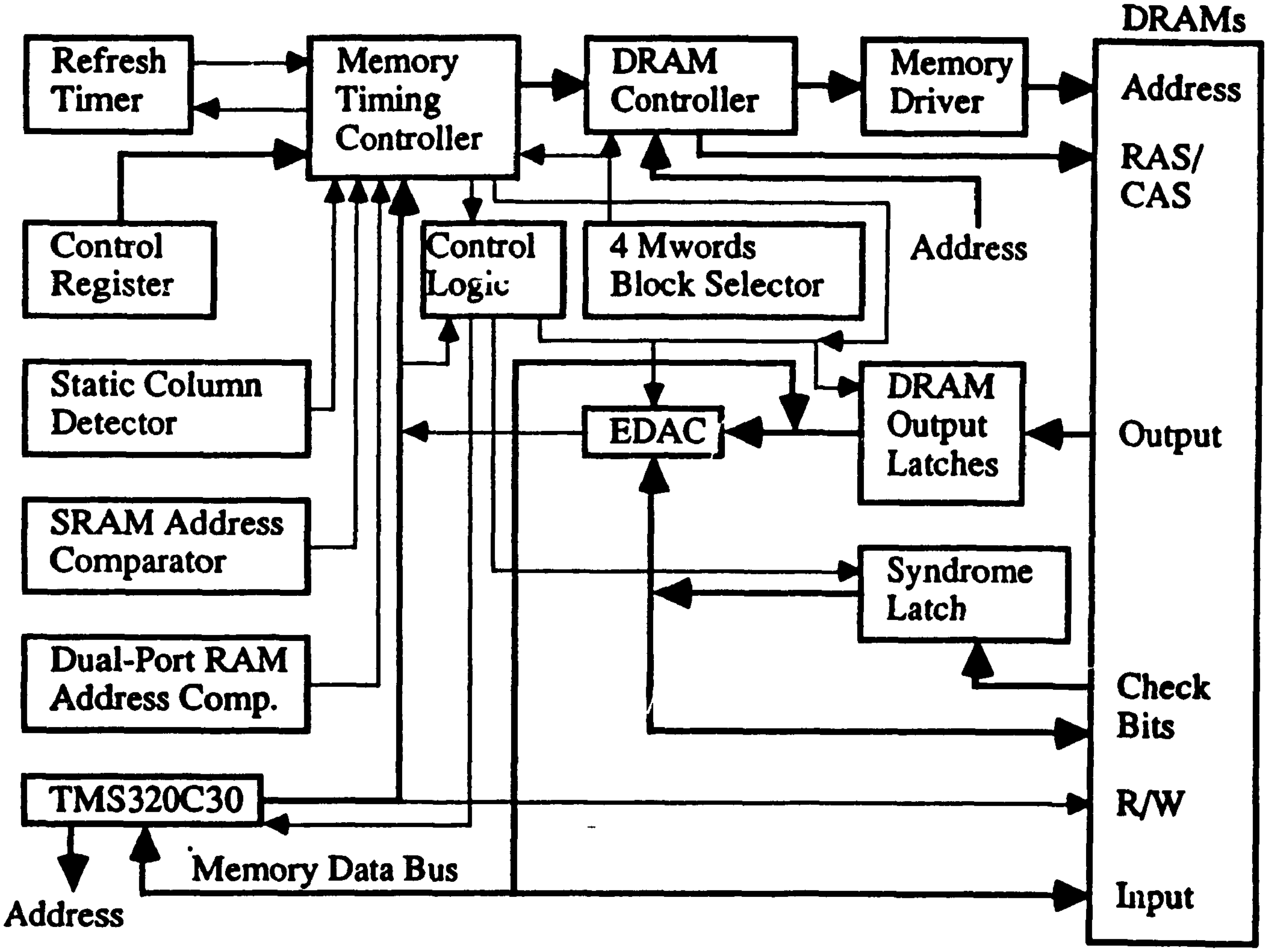


Figure 3.1 - Block diagram of the DRAM interface.

CHAPTER THREE - MEMORY

The 1Mx1 bit DRAM can be refreshed in three ways: RAS (row address strobe) only refresh, CAS (column address strobe) before RAS refresh, and hidden refresh. The hidden refresh permits the data to be valid at the output during refresh. Since the DRAM interface uses DRAM output latches, hidden refresh is not considered. RAS only refresh has been adopted since it is the easiest to implement. It requires a row counter which is included in many DRAM controllers [9]. Memory scrubbing [10] has not been implemented because of the complexity in the design involved compared to the improvement in the fault tolerance. The flow-through policy has been adopted instead.

The refresh timer is based on a 22V10 PAL [11]. The SN74ALS6300 input selectable refresh timer from Texas Instruments [12] could have been selected but we felt that the PAL solution offered more flexibility. The device is driven by H1, a clock provided by the TMS320C30 that runs at 16.5 MHz. The 33 MHz system clock has not been selected in order to simplify the program for the PAL and because the H1 clock provides a better timing reference for the memory accesses. The device sends a REFREQ (REFresh REQuested) signal every 160 clock cycles. The division rate of 160 (which should be adjusted with the DRAM interface used) provides extra time for timing arbitration in the memory timing controller. When the refresh is complete, the memory timing controller sends a RFC (Refresh Complete) signal back to the refresh timer.

3.1.2 Memory timing controller

A programmable logic sequencer such as the PLS105 [11] is configured as a state machine. The timing controller is initialized by taking the reset input low. From the initialization state (state 0), the timing controller can perform either an access, a refresh cycle, or remain at state 0 depending on the signals STRB, DP, STATIC, H1, and REFREQ. STRB is active when the processor accesses the DRAMs, the SRAMs, the dual-port RAMs, or the expansion board. The timing controller assumes that it is a DRAM access when STRB is active and if DP and STATIC (the dual-port and SRAM access strobes respectively) are not active. A bit (MEM) is provided in an external register (DRAM control register) to indicate to the timing controller that the expansion board is used as memory. The refresh timer is synchronized by H1.

The timing controller performs the refresh cycle, requested by the refresh timer through the REFREQ signal, immediately if the TMS320C30 is not doing a DRAM access. If the controller is performing a DRAM access, the refresh cycle will be delayed until the access cycle is complete. If the controller is asked to perform an access cycle during a refresh. the access cycle will begin immediately after the refresh cycle is complete.

To indicate to the DRAM controller which task to perform, the memory timing controller sends four signals which will be explained in the next section. It also controls the latch enable (LATCH) of the DRAM output and syndrome latches. as well as a part of the EDAC. mainly to enable (OEDATA) the correct value after the first successive read in the DRAM address after the ERR (error) flag has been active.

The memory timing controller can support two DRAM portions: A 4 Mword DRAM part and a 12 Mword DRAM part situated on the expansion board at the end of the connection bus. A 4 Mword block selector indicates to the timing controller through SELECT that the 4 Mword block is accessed. The 4 Mword block selector permits allocation of the extended memory to any of the four positions in the 16 Mword memory map. The optional extension of up to 12 Mwords of DRAMs indicated by MEM can have the same or a different number of wait-states when accessed. The DRAM control register provides two lines, namely WS0 and WS1, which indicate to the timing controller the number of extra wait-states relative to the 4 Mword DRAM part, that must be added when the extension DRAMs (SELECT high) are accessed. When both lines are low, the extension memory is accessed at the same speed as the 4 Mword part. Up to 180 ns can be added to the access time.

The memory timing controller provides the timing requirements in accessing the DRAMs. Because a hierarchical memory structure can be adopted with a large slow access (100 ns) DRAM and a relatively small fast access (25 ns) SRAM blocks. fast DMA transfers must be performed between the two levels to ensure that the instructions and data words are available soon enough to guarantee no wait-state access cycles. Three types of DRAM access are available beside the normal access mode. namely the page. nibble. and static column decode modes. They all provide high-speed access to the DRAMs. Depending on the mode selected, a different 1Mx1 bit device must be chosen.

Nibble mode allows the highest rate by cycling the CAS clock while holding the RAS clock active. Internal row and column address counters increment at each CAS cycle. thus no external column addresses are required. unlike the other modes. After cycling CAS three times in nibble mode. the address sequence repeats and the same four bits are accessed. in serial order. upon subsequent cycles of CAS.

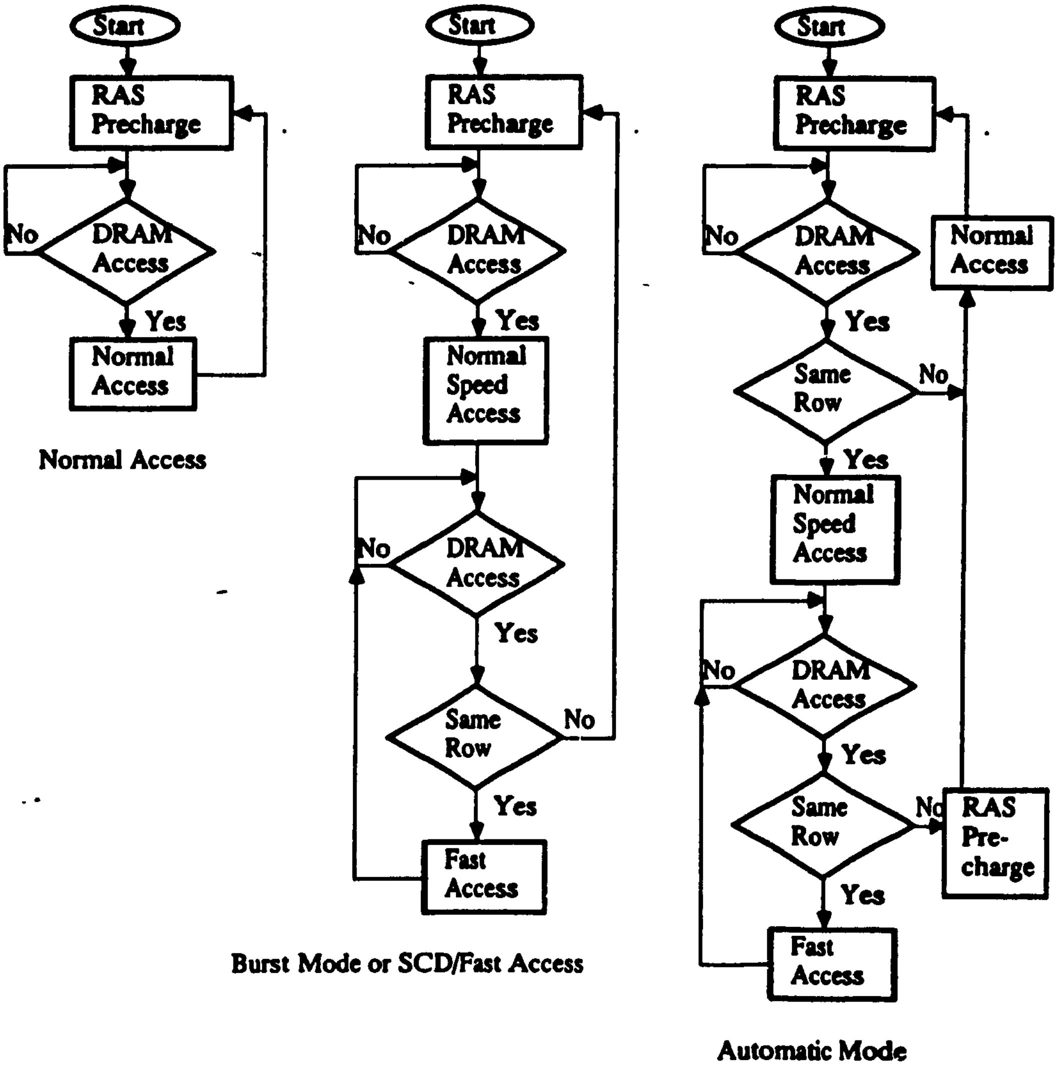


Figure 3.2 - The three selectable DRAM access modes.

Page and static column modes allow access to any of 1024 column locations on a specific row. In both modes, the RAS signal is held low. The CAS signal fluctuates between high and low for each access in page mode while it is held low in static column mode resulting in less noise.

States	MC1	MSEL	RASI	CASI	READY	RFC	LATCH	OEDATA
ST0	HIGH	LOW	HIGH	HIGH	HIGH	LOW	LOW	HIGH
ST1	HIGH	LOW	LOW	HIGH	HIGH	LOW	LOW	HIGH
ST2	HIGH	HIGH	LOW	LOW	LOW	LOW	HIGH	HIGH
ST3	HIGH	LOW	HIGH	HIGH	HIGH	LOW	LOW	HIGH
ST4-5	LOW	LOW	LOW	HIGH	HIGH	HIGH	LOW	HIGH
ST6	HIGH	LOW	HIGH	HIGH	HIGH	LOW	LOW	HIGH
ST7	HIGH	HIGH	LOW	LOW	LOW	LOW	HIGH	HIGH
ST8-14	HIGH	HIGH	LOW	LOW	HIGH	LOW	LOW	HIGH
ST15	HIGH	LOW	HIGH	HIGH	LOW	LOW	LOW	LOW
ST16	HIGH	HIGH	LOW	LOW	LOW	LOW	LOW	LOW
Active			LOW	LOW	LOW	HIGH	HIGH	LOW

TABLE 1 - Logic Levels of the Signals Generated by the Memory Timing Controller.

* Note: It might be necessary to add states to compensate for the propagation delay through the DRAM interface.

Since the nibble mode can only access a maximum of four bits, and because of the noise problems involved in page mode, static column decode has been adopted. To perform static column decode [13], the row address must be checked. A 74ALS6310 static column and page mode detector [14] compares the new with the last DRAM row address and indicates to the timing controller through the HSA signal if a fast access can be performed. While in the fast access mode (static column), the memory timing controller assumes that the next DRAM access will be a fast access and keeps both RAS and CAS low. Since the access cycle is reduced by about half in the fast access mode (burst mode), a penalty of one extra wait-state compared to the normal access mode is imposed when leaving this SCD mode due to the RAS precharge time. The timing controller might leave the fast access mode

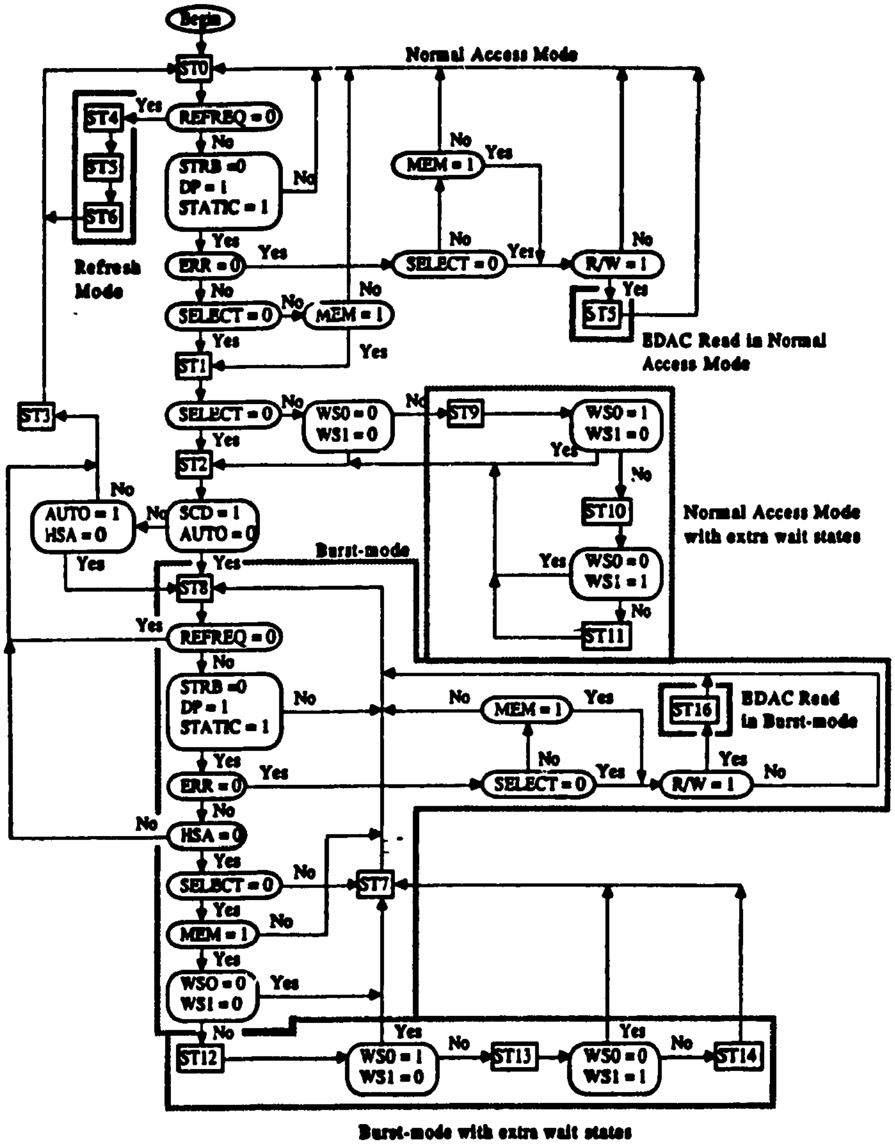


Figure 3.3 - Flowchart of the different operations of the timing controller.

when the row address is not the same as the previous DRAM access or when a refresh must be performed. It is thus better suited to perform normal accesses only when random accesses on DRAMs are predicted. A static column decode switch is provided in the DRAM control register to disable high-speed accesses. To prevent the programmer from switching from normal to fast access. by changing the SCD switch frequently. an automatic mode (AUTO) is provided to allow the system to switch between both modes without the user's intervention. Figure 3.2 shows the difference between the three DRAM access modes.

For more clarity. the flowchart in Figure 3.3 indicates the different operations of the timing controller. Table 1 shows the logic level of outputs MC1. MSEL. RASI. CASI. READY. RFC, LATCH. and OEDATA for the different states.

CASI for states 8 and 16 could be changed to "1" to adapt the system to page mode. The four signals MC1. MSEL. RASI, and CASI are explained in the DRAM controller section. The READY is connected to the processor via an AND gate. During a DRAM access. the TMS320C30 will wait until READY is low.

3.1.3 Dynamic RAM controller

The DRAM controller is based on the TI 74ALS6301 [15]. This device is capable of controlling any DRAM up to 1M bits. It typically operates in a read/write or a refresh mode. During normal read/write operations. the two signals RASI and CASI are activated by the memory timing controller to strobe the row and column addresses which are multiplexed by the DRAM controller and sent to the DRAMs. MSEL from the timing controller selects the row or column address. In this mode, the timing controller keeps MC1 high. MC0 is tied to ground making refresh with scrubbing and clearing the refresh counters impossible to perform. Since RAS only refresh is considered. a transition of MC1 to a low level is sufficient to perform the refresh without scrubbing. Refresh cycles are conducted using the internal row counter in the DRAM controller to generate the addresses. In this mode all RAS outputs are active while the four CAS outputs remain high. Since one DRAM controller can be interfaced to four memory banks of 1 Mword each. address lines AB20-21 connected to SEL0 and SEL1 determine which RASn and CASn are activated. and thus which bank is accessed. With the 74ALS6301, the refresh counter is incremented at the rising edge of RASI. Since the TMS320C30 processor has separate address and data buses. the input latches are left tranparent by tying the latch enable (LE) input at Vcc.

3.1.4 Memory driver

All outputs of the DRAM controller can drive up to 12 mA (I_{o1}). Because each RAS and CAS lines drive a maximum of 1 Mword while the Q outputs (address lines) drive 4 Mwords of memory. to maintain maximum performance in a 32-bit system with parity. a driver is provided to the Q outputs to give the extra drive up to four times I_{o1} of the RAS or CAS outputs. One good device is the BCT2828 [16] with I_{o1} = 48 mA using the BICMOS technology which lowers power consumption and reduces undershoot inherent in high capacitive load configuration. It also provides ten outputs and is thus well suited for the multiplexed row and column addresses of the 1Mx1 bit DRAMs.

3.1.5 DRAM outputs and syndrome latches

DRAMs have separate input and output pins. To guarantee that the data outputs are available to the TMS320C30, the data outputs are latched and remain valid until the following DRAM read cycle.

For the syndrome latch. the strategy is the same except that the 7-bit code used for EDAC applications is retained and is available exclusively to the EDAC. The latch enable as for the DRAM output latches is controlled by LATCH from the memory timing controller, while the output enable for both configurations is controlled by the timing controller and the bus/EDAC controller.

3.1.6 DRAM access time

An important task to be performed in the design phase is to determine the access time required for the dynamic RAMs. In general. memory contributes the greatest cost to a computer system. The cost of the system can be significantly reduced if the DRAMs are chosen with the most appropriate access time.

CHAPTER THREE - MEMORY

In a high performance system, it is necessary to minimize the number of wait-states associated with each DRAM access. Today, one of the fastest 1Mx1 bit DRAM has an access time of 85 ns [17]. Since the TMS320C30 requires an access time of 35 ns for zero wait-state, generating wait-states must be considered in accessing DRAMs. Each wait-state adds another 60 ns to the access time. With one wait-state, the access time should be less than 95 ns allowing 10 ns for all delays generated by the DRAM interface. This requirement is impossible to meet, thus more than one wait-state must be considered. Up to five wait-states could be used in some cases in the normal access mode, or one to two wait-states in the burst mode. In choosing the right DRAM, all access modes implemented must be considered. The propagation delays through the DRAM interface, as well as the clock periods necessary for the memory timing controller to generate the appropriate signals, must also be considered. The propagation delay in a highly capacitive environment on the connection bus and the interfaces must also be taken into account. When the worst case propagation delays have been considered, a different value (RESULT) can be used for each access mode to determine the best DRAM. The following simple equation can then be applied to determine the DRAM access time required (t_{acc}):

$t_{acc} < 60W - (RESULT - 35)$ where W is the number of wait-states.

3.1.7 Error detection and correction

For system memory sizes larger than 0.5 Mbits, the MTBF (Mean Time Between Failures) is significantly reduced [8]. This is usually due to soft errors which are random memory value changes (usually from a high to a low level). These errors may be caused by system noise, alpha particle radiation, or power surges. For 1Mx1 bit DRAMs, the density of memory chips increases their probability of errors. Therefore, data integrity decreases in larger memory arrays. For a 1 Mbit chip, a typical scft-error rate of 0.20 - 0.35 % per 1000 hours can be expected [8]. Since 32 such devices are needed to form 1 Mword, the typical soft-error rate becomes 6.4 - 11.2 % per 1000 hours. For 4 Mwords of local memory, this becomes 25.6 - 44.8 % per 1000 hours. This is roughly one soft-error every 3000 hours. For the whole system with 31 local memories of 4 Mwords each, the typical soft-error rate increases to one soft-error for each 72 - 126 hours of operations. With such results, it becomes necessary to perform error detection. Two options are available to the designer. The first alternative is to perform error detection only. This has the advantage of being easily accomplished by using parity bus transceivers. The detection time is very fast. Also fewer DRAMs are necessary to store the parity bits. However if an error is detected. the system (or part of it) must be rebooted. and all data in the RAMs would be lost. In order to extend the MTBF, an error-correction scheme must be incorporated. An EDAC device such as the 74AS632 [18] may be considered to implement this second option. The disadvantages are that it is more difficult to implement. the detection time is greater than parity bus transceivers. and more DRAMs are required (seven per 1 Mword) to hold the parity bits or check bits necessary to form the modified Hamming code [19]. Since we conduct experiments with a duration exceeding 72 - 126 hours. involving all processors. it is apparent that error-correction is necessary.

To avoid waisting crucial time in DRAM read cycles (see DRAM access time) by adding the EDAC detection and correction time, an EDAC bypass read access has been adopted. This means that during a read access period. the data are latched and available to the EDAC and to the DSP. The processor then assumes that the data are valid. During the remainder of the read access cycle, the EDAC performs the detection and correction phases. which are completed before the beginning of the next cycle. If an error is detected, the ERR flag from the EDAC generates a high-priority interrupt. The processor then accesses the correct value available on the primary bus through the interrupt routine. MERR indicating multiple errors and involving a partial or full rebooting of the system, can be checked in this routine. Also the software option to error detection and correction adds flexibility. For example. the programmer can choose to write back the correct value to the DRAMs. thus minimizing the risk of obtaining multiple errors. For a "write". the write cycle being 60 ns longer than the read cycle. sufficient time is allowed to the EDAC to generate the syndrome bits.

Since the risk of an error is very low in a DRAM read access, it is worthwhile adopting the EDAC bypass read access. This minimizes the cost of the system by allowing the designer to choose slower DRAMs. Also, despite the fact that the EDAC bypass read access is slower than Read-Flag-Correct or Read-

Modify-Write operations when an error has occured, it is faster or more economical when no error is detected. Thus in our situation, the EDAC bypass read access is a better choice.

3.2 Static RAM block

Up to 16Mx32 of memory can be assigned to each processor. Because dynamic RAMs impose wait-states causing a degradation of the performance, static RAMs must be considered to optimize fully the throughput of the system.

Special techniques can reduce significantly the number of wait-states associated with a DRAM-based memory. Sophisticated access modes such as nibble, page, and static column improve the average access time to the memory but do not guarantee a zero wait-state memory access especially for a processor running at 33 MHz. A cache [20] decreases the number of accesses to slower memory but does not eliminate completely the wait-states. DSPs are often used in real-time applications and despite the fact that a sophisticated cache system is implemented, a worst case access time should be considered since the hit rate is hard to predict and time is critical. Also, the goal in designing a cache is to obtain zero wait-state accesses when a hit occurs, and to maximize the hit rate. The hit rate will rarely reach the 100 % hit ratio because it is based on a concept known as locality of reference. In the case where a large amount of data (larger than the size of the cache) are required, a large miss rate can be expected, thus degrading significantly the overall system performance. This is one example of the problems involved in a cache-based design where applications such as image processing must deal with large block of data. Another problem with the external cache is that current technology does not provide a cache tag (required for determining a hit or a miss) fast enough for the TMS320C30. A fast cache tag reflects a worst case comparaison time of 25 ns [21] which is too long with respect to the maximum allowed time to indicate to the DSP if a hit occured for a zero-wait state access. Another major problem would be the cache-reload transient described in [22]. In extensive multi-tasking systems, transitions from one process to another involve high rates of cache flush thus increasing the miss ratio.

For relatively large memory, both static and dynamic devices should be considered. The dynamic RAMs due to their small package, high density, low power consumption, and low price, would contain most of the instructions and data. The TMS320C30 has 2Kx32 bits of internal RAM. Since this might not be enough for most applications, a larger external memory board with zero wait-state is necessary. The resulting hierarchical memory structure would not reduce the computing speed of the DSP since an internal bus in the TMS320C30 is dedicated for DMA transfers only. The following describes the steps in designing such a memory board by applying static and dynamic analysis to predict the behaviour of the system.

3.2.1 Choice of the bus

This fast local memory could be accessed by either the primary bus or the secondary bus. Up to 8Kx32 bits of memory can be interfaced through the secondary bus. Since we intend to assign more than 8Kx32 bits of external zero wait-state memory, the primary bus has been selected for all accesses to the fast local memory.

3.2.2 Access time for the static RAM

SRAM devices are available on different versions and access times. Typically, access times increase by 10 ns between versions. The typical relatively high-density, high-speed SRAMs are offered with access times of 15, 25, 35, and 45 ns. Only 35 ns are available from the address valid to the data valid, to perform a zero wait-state memory access with the TMS320C30. In order to obtain the best board density, the slowest possible device (i.e. highest density) should be selected (but keeping in mind that delays will be generated by the memory interface). The 25 ns SRAM seems to be the best choice since it is possible but not easy to deal with 10 ns for decoding and buffering.

3.2.3 Static RAM configuration

Due to several reasons such as the price, board space, and capacitance, the total capacity of the board has been restricted to 2 Mbytes. Today, the highest density static RAMs with a 25 ns access time, have the 256Kx1, 64Kx4, or the 32Kx8 bit configurations [23]. One important aspect to consider in selecting these devices is the power consumption. CMOS SRAMs in this category and offering low active and standby power are available. They provide an automatic power-down when de-selected. In order to take advantage of this feature, the designer must divide the whole static RAM memory in many banks, where only one bank can be accessed at a time. With the 256Kx1 bit SRAMs, two banks would be required and half of the whole memory would be selected when the board is accessed, increasing considerably the power consumption. 8 and 16 memory banks are required when the 64Kx4 and the 32Kx8 bit devices are used respectively. For reasons described later concerning particularly the decoding scheme that should be kept efficient, the 64Kx4 bit configuration is best suited for our case.

3.2.4 Bank switching

The TMS320C30 offers a programmable bank switching feature which allows the processor to insert automatically a wait-state when a different bank is accessed. This permits the designer to opt for slower decode circuitry. This works fine especially when the same type of memory is being used. In our particular case, dual-port and dynamic RAMs share the same bus. Because the size of the bank of these memories is different from the SRAM bank, it makes it difficult to operate efficiently with the bank switching approach. The bank size would need to be adjusted frequently in such a configuration. The most important aspect to consider are the DMA tranfers which are likely to be used often in our system. DMA transfers will typically occur between the internal RAM, the external zero wait-state memory, the dual-port RAMs, and the DRAMs. Let us assume a simple DMA transfer between two banks of our zero wait-state memory board. In order to optimize fully the bank switching, the internal RAM would first be loaded from the source memory bank. Secondly, a second DMA transfer would be required to unload the internal RAM to the destination bank. The first transfer could reflect a maximum rate of 22.2 Mbytes/s and 33.3 Mbytes/s for the second DMA transfer resulting in 13.9 Mbytes/s. This result is the same when bank switching is used or when a fast decoding scheme is provided. Using one DMA transfer may result in a maximum rate of 16.7 Mbytes/s for the true zero wait-state scheme with no wait-state between bank accesses, or 13.3 Mbytes/s for the bank switching option.

An attempt to design a true zero-wait state memory system by providing a very fast memory interface is worthwhile in our case due to an intensive use of DMA transfers.

3.2.5 Decode section

Since less than 10 ns are available to decode the 24 address lines, the decode section represents a good challenge for the designer. From the memory address bus, 16 lines are forwarded to the static RAMs to select a specific address in a particular bank. The next three address lines selects the respective bank, and the remainding five most significant lines enable the memory board.

To optimize the speed, fast ICs must be selected. The FCT devices from IDT (Integrated Device Technology) are very fast, have a good fan-out, and are both TTL and CMOS levels compatible. Figure 3.4 shows a typical decoding scheme. A comparator is used to enable a bank selector. This scheme has the advantage that it is very simple but it uses a two-level decode. In this two-level circuit, the propagation delay through the 74FCT521B address comparator, and the 74FCT138A decoder, must be added together resulting in a maximum propagation delay of 11.4 ns for the decode part only. Transceivers must also be used for the data due to the high capacitance imposed by large memory. The 74FCT245A or the 74FCT645A represents another 4.6 ns. Thus only 5.4 ns are allowed in the worst case for the decode section. This eliminates the first option.

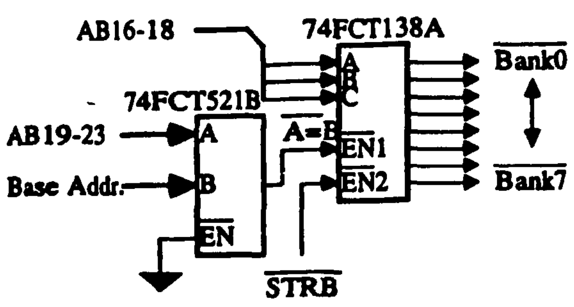


Figure 3.4 - A standard decoding scheme.

One solution is to choose the 15 ns SRAMs, but 256 instead of 64 static RAM ICs might be required to populate fullythe 2 Mbyte board since the 64Kx1 bit configuration has one of the highest densities for the 15 ns version. Another approach is to reduce the decode circuitry to one level by doing the board and the bank selects in parallel. Figure 3.5 shows this second alternative.

This solution requires SRAMs with two chip-selects. Notice that the market offers two main types of static RAMs, one has only one chip-select, and the other provides two chip-selects and an output enable pin. The first type is available in a 24-pin skinny DIP package and the second requires 28 pins resulting in a larger board layout. The solution is to select the PLCC or other similar packages. But another problem persists. The output of the 8-bit comparator must be connected to 64 SRAM chip enable pins resulting in a high capacitive load at the end of the output pin. The maximum propagation delay of 5.5 ns for the 74FCT521B is specified for a 50 pF load. In order to maintain a maximum performance through the comparator. up to eight 74FCT521Bs might be required.

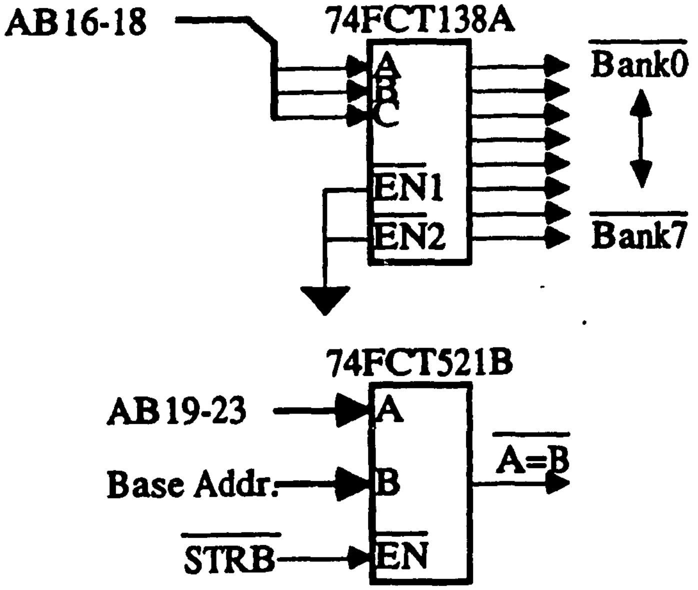


Figure 3.5 - A one-level decoding scheme requiring two chip enables.

Another alternative is to provide an address comparator for each bank as shown in Figure 3.6. This solution requires SRAMs with only one chip-select. The decoder of the previous altcmative is eliminated and all inputs of the comparators are used for full optimization of the IC. Many SRAMs reflecting the required characteristics are available on the market. Typically they are represented with a maximum input capacitance of 5 pF for each pin. Since eight 64Kx4 bit SRAMs are required for each bank, a maximum of 40 pF may be connected to one output of a bank selector. This is well below 50 pF if we consider that all SRAMs in the same bank are unlikely to show the maximum but rather the typical value.

To respect the memory access timing of the TMS320C30, the capacitance at each output should be kept as low as possible. The typical input capacitance of the 74FCT521B is 6 pF. Since eight such devices are connected to AB16-23 and STRB (being the eight most significant address lines and the memory access strobe on the primary bus respectively), a typical C_{in} (input capacitance) of 48 pF will not modify the timing characteristics of the DSP. With the 32Kx8 bit SRAMs, C_{in} could increase to a typical value of 96 pF which is too high for the TMS320C30 to drive efficiently the decode circuitry. In this case, drivers would be required between the DSP and the comparators, adding an extra stage and increasing too much the total propagation delay of the memory interface.

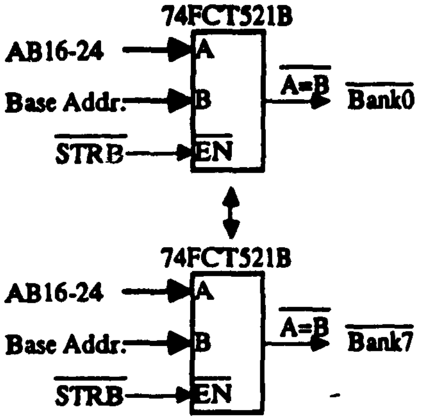


Figure 3.6 - A one-level decoding scheme requiring one chip enable.

3.2.6 Number of address drivers

Since the address lines AB0-15 are forwarded to all SRAMs in parallel with the bank selects, a propagation delay less than or equal to the propagation delay through the decode circuitry must be maintained Two 74FCT244As with a maximum propagation delay of 4.4 ns for 50 pF load, are required to drive the 16 least significant address lines. If only one pair of 74FCT244As is used to drive the eight banks, then a maximum of 320 pF (8 SRAMs per bank * 8 banks * 5 pF) could be tied at each output of the address drivers. IDT recommends to add a maximum of 3 ns to the propagation delay for each 100 pF above 50 pF of capacitive load for FCT devices. Thus the following equation will determine the new propagation delay:

$$t'_{pd} = t_{pd} + (C_l - 50) * 3 / 100.$$

t'_{pd} and t_{pd} are the new and the old propagation delay in ns respectively. C_l is the total capacitive load in pF connected to the output. With one pair of drivers, the new propagation delay equals 12.4 ns which is significantly larger than the propagation delay of 5.5 ns of the SRAM decode circuit. Doubling the number of drivers will significantly reduce the capacitive load at each output, where each set of drivers is connected to only half of the entire memory array. In this case, t'_{pd} is 7.6 ns maximum. This is slightly more than the tolerable value. Using one pair of drivers for every two banks, results in a maximum propagation delay of 5.2 ns. Thus eight 74FCT244As should be considered to drive AB0-15 in order to maintain in all cases the high throughput of the system. With this configuration, typical and maximum capacitive loads of 24 and 40 pF would be tied to each AB0-15 pin of the TMS320C30. Another advantage of using many drivers is that damping resistors are not required since the capacitance is quite low.

3.2.7 Number of transceivers

Four 8-bit transceivers such as the 74FCT245A or the 74FCT645A are necessary to drive a 32-bit word. The 74FCT645A is preferred since it has the same characteristics as the 74FCT245A but has all inputs on one side and all outputs on the other side, simplifying the PCB layout. Each I/O pin of a typical SRAM is characterized at 7 pF maximum. Thus for all eight memory banks, a maximum of 56 pF is connected to the pin of the transceiver. This does not cause any problem when the tranceivers drive the SRAMs during a "write" operation. For a "read", the driving sources become the SRAMs which reflect less driving capability.

For typical CMOS SRAMs from IDT or Cypress, the characteristics of the device are given for 30 pF. Up to 6 ns for each additional 100 pF above 30 pF could be added to the normal access time of the SRAM when driving higher capacitance. Thus the following equation should be used in this case:

$$t'_{acc} = t_{acc} + (C_l - 30) * 6 / 100.$$

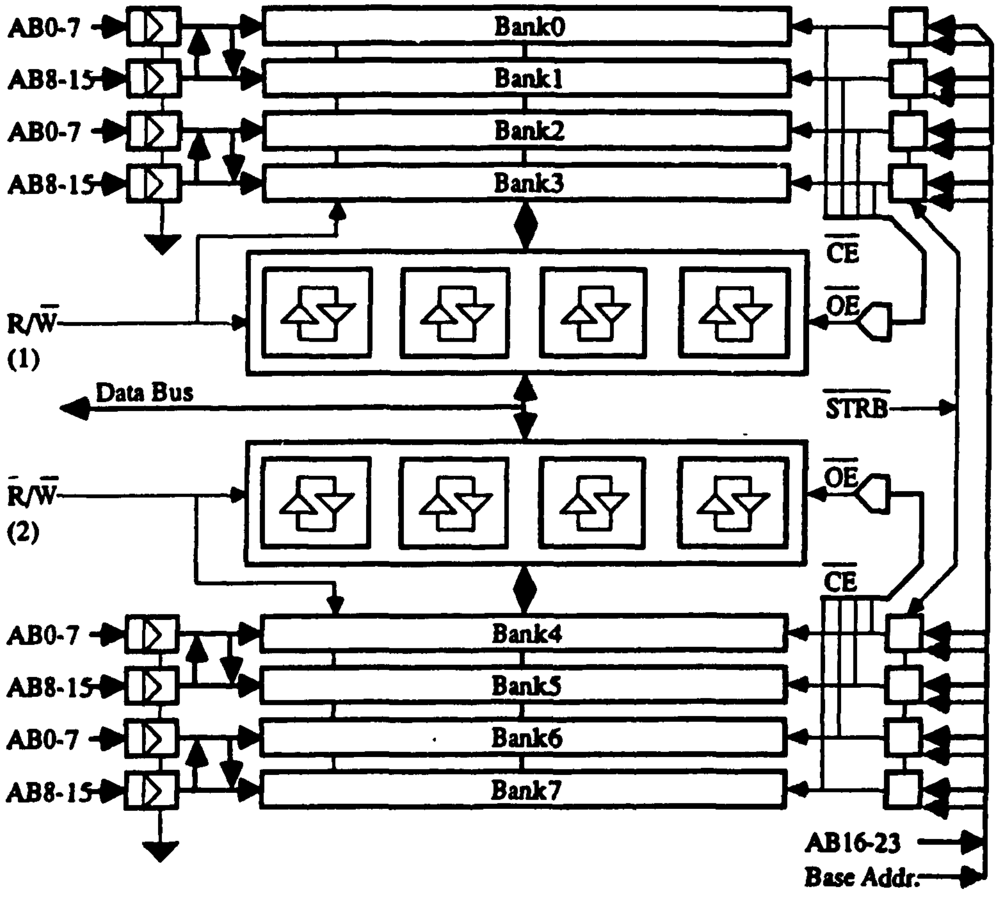


Figure 3.7 - Block diagram of the 512Kx32 bit memory board.

t'_{acc} and t_{acc} are the new and the old access times in ns respectively. One SRAM output might drive seven other SRAM data pins and one I/O pin of a transceiver. The maximum $C_{i/o}$ (input/output capacitance) of the 74FCT645A is 10 pF, thus for 59 pF (10+7x7), t'_{acc} = 26.7 ns. Since we cannot afford to waste even 2 ns. an attempt should be made to optimize the access time of the SRAMs.

By doubling the number of transceivers, $t'_{acc} = t_{acc}$ = 25 ns maximum since the maximum capacitance equals 31 pF which is very closed to the 30 pF specified on the data sheets. Thus eight tranceivers should be considered. Figure 3.7 shows the block diagram of the 512Kx32 bit SRAM memory board.

3.2.8 Transceivers output enable control

As seen in Figure 3.7, two 4-input AND gates are required to control the output enables of the transceivers. Each gate is controlled by four bank strobes. Since no FCT type gates are available, TTL or CMOS gates must be considered. The 7421 chip has two 4-input positive-AND gates well suited for our application. Since only a maximum of 40 pF with a very low fan-in would be connected to the output of each gate, a driver is not necessary.

To determine the maximum tolerable propagation delay for the gates, we must consider both the "read" and the "write" operations. For a "read", the propagation delay must not be greater than the access time of the static RAMs less the output enable time of the 74FCT645A which results in 18.7 ns. For a "write", the timing diagrams of the TMS320C30 and the SRAM used must be considered. With the access timing characteristics of the TMS320C30, write cycles with chip enable controlled must be used to determine the timing values of the SRAMs. The data set-up to write end (end of access strobe) of most 25 ns 64Kx4 bit SRAMs is 10 ns minimum. The width of the access strobe STRB is 60 ns. Thus the maximum propagation delay from high to low must be less than 60 - (t_{pzh} + t_{sd}) where t_{sd} is the SRAM set-up time and t_{pzh} is the output enable time for the 74FCT645A which is 6.2 ns. Thus t_{phl} for the 7421 must be less than 43.8 ns in this case. The "read" access represents the worst situation with only 18.7 ns to enable the transceivers. The 74HC21. the CMOS version. has a maximum propagation delay of 28 ns at 4.5 V and C_l =50 pF. This is too slow. Since both TTL and CMOS can be interfaced with FCT devices, the 74ALS21A may be considered despite a small increase in the power consumption. It has a maximum propagation delay from high to low of 10 ns which is less than 18.7 ns, and a maximum propagation delay fron low to high of 15 ns which is less than the time between two consecutive accesses. These values are given for Vcc=4.5 to 5.5 V, and C_l =50 pF.

3.2.9 Read/write line

The R/W line is connected to 64 static RAMs and 8 transceivers, resulting in a possible maximum capacitive load of 400 pF. According to the timing diagram of the SRAMs, the write enable (WE) pin connected to the R/W line is allowed to change states with the chip enable (CE) pin simultaneously. Observing the timing diagram of the TMS320C30 reveals that the R/W line goes low 30 ns before STRB used to generate the chip enable. The propagation delay from STRB to CE is 4.6 ns maximum (see 74FCT521B data sheets). Thus 34.6 ns can be used as an acceptable maximum propagation delay for the R/W. Since an FCT driver is used on the DSP board to drive the R/W line, the equation is Section 3.2.6 is sufficient to perform the dynamic analysis. With 400 pF, the maximum propagation delay increases to 14.8 ns which is acceptable.

As far as the static analysis is concerned, the input load current (I_{ix}) of the CMOS SRAMs must be considered. For the CY7C194 foı example, I_{ix} is -10 uA minimum and +10 uA maximum. For the 74FCT645A transceiver, the input high current (I_{ih}) is +5 uA maximum and the input low current (I_{il}) is -5 uA maximum. One output of the 74FCT244A can source up to 64 mA for logic zero (I_{ol}), and sinks up to -15 mA for logic one (I_{oh}). For a high logic level on the R/W line, the total input high current is 680 uA maximum, and a total input low current at a low logic of -68 uA. In both situations, the static analysis is successful.

But this analysis is for TTL levels for a guaranteed logic high level of 2.0 V minimum and a maximum of 0.8 V for a guaranteed logic low level. In order to bring the lines to CMOS levels and minimize the power consumption, I_{oh} must be reduced to -300 uA and I_{ol} to +300 uA. Since the maximum total input high and low current is plus or minus 680 uA, then a separate R/W line should be dedicated to each 4 banks and transceivers. The new capacitive load seen by each R/W line would be reduced to a maximum possible value of 200 pF resulting in a t'_{pd} of 8.8 ns.

3.2.10 Total access time

To determine the total maximum access time of the SRAM memory board, the propagation delay through the comparators and address drivers, the maximum access time of the static RAMs, and the propagation delay through the transceivers must be taken into account. As calculated in the previous sections, 5.5 ns maximum are required for the decode and address drivers, another 25 ns are needed to access the SRAMs, and 4.6 ns are required to propagate through the transceivers. Adding these values results in a worst case total access time of 35.1 ns.

3.2.11 Memory re-allocation

Since a memory size of 512 Kwords can fit in many different portions of the 16 Mwords memory map of the TMS320C30, it becomes possible to add a feature requiring zero glue-logic on the memory board to allow the programmer to re-allocate the SRAM memory into the memory map. This feature adds flexibility and makes it compatible with a variety of configurations. As seen in Figure 3.7, the eight most significant address lines on the primary bus are used to enable the proper bank. AB16 to AB18 could be used to decode a particular memory bank, and AB19 to AB23 to re-allocate the memory board. This re-allocation feature would only require five supplemental lines on the connector linking the memory board to the processor board. These lines could be driven by a portion of an external register under software control and situated on the DSP board.

The bank selectors implemented with 8-bit comparators do both the bank and the board selection. Each comparator has A and B entries. When the 8-bit input of port A connected to AB16-23 are equal to a base address on the port B, a bank access strobe is activated. On the port B, only the five most significant inputs are connected to the portion of the external register containing the base address. The three least significant pins of port B would be connected to ground or +Vcc through a pull-up resistor in order to cover all binary combinations from 000 to 111 where a particular combination is assigned to a particular bank.

3.2.12 PAL-decode approach

It is possible to replace all 74FCT521B 8-bit comparators by one 16L8 PAL. Figure 3.8 shows the configuration:

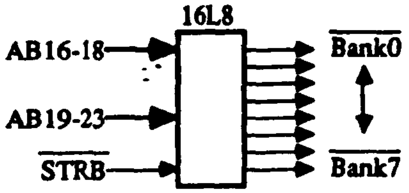


Figure 3.8 - A PAL-based decoding scheme.

A relatively low capacitance is connected to each output of the PAL. The chip count is also reduced. But, since not enough inputs and product terms are available, the memory re-allocation feature cannot be implemented here. With a fixed base address, only one product term is required for each bank access strobe. The following equations would be programmed in the PAL to implement the decoder for a memory board being accessed at the end of the memory map:

```
/BANK0 = /AB16*/AB17*/AB18*AB19*AB20*AB21*AB22*AB23*/STRB
/BANK1 = AB16*/AB17*/AB18*AB19*AB20*AB21*AB22*AB23*/STRB
/BANK2 = /AB16*AB17*/AB18*AB19*AB20*AB21*AB22*AB23*/STRB
/BANK3 = AB16*AB17*/AB18*AB19*AB20*AB21*AB22*AB23*/STRB
/BANK4 = /AB16*/AB17*AB18*AB19*AB20*AB21*AB22*AB23*/STRB
/BANK5 = AB16*/AB17*AB18*AB19*AB20*AB21*AB22*AB23*/STRB
/BANK6 = /AB16*AB17*AB18*AB19*AB20*AB21*AB22*AB23*/STRB
/BANK7 = AB16*AB17*AB18*AB19*AB20*AB21*AB22*AB23*/STRB
```

Programming the PAL as a decoder is thus a very simple task. The last difficulty is to find a PAL with a maximum propagation delay of 5.5 ns corresponding to the maximum propagation delay of the decoder of the last alternative. Today, the fastest PAL of this type refects a maximum propagation delay of 7 ns.

The PAL-based decoder becomes then interresting when the number of chips must be minimized. the decoding speed is not critical, and the re-allocation feature is replaced by a fixed and less flexible access configuration.

CHAPTER FOUR

MAIN COMPONENTS OF THE DSP BOARD

4.1 Multicast system

The overhead associated with performing a communication has a significant influence on the performance of an application. Let us define R and W as read and write access times respectively. In a typical single bus multiprocessor system, if one processor wishes to transfer a word to n other processors via a mailbox in a shared main memory. 1W + nR is required to complete the transfer. The bottleneck here is due to the fact that only one read cycle can be performed at a time without contention. In our system. due to the fact that each processor has its own bus. n read accesses can be performed simultaneously. To provide the same performance for the write cycles. the processor must be able to write to different dual-port modules simultaneously.

To avoid the drawback of the unicast option, some systems use a broadcast approach such as sending a message via tne bus with a set identifier. In this case. all DSPs must be interrupted to verify if they are to receive the message. Since multiple transfers are often required. a high throughput is fairly difficult to achieve in this case. One option is to attach a communication processor to each DSP, which will check the identifier and interrupt the DSP as required. This approach has some advantages but complicates the design of the processor board as well as limiting the transfer rate between all processors because of bus contention problems.

Another approach is to interrupt only the receiving DSPs. It is referred to as multicast (one-to-many transmissions) [24]. This technique, used with the novel architecture based on dual-port memory modules and multiple buses, corrects the problems encountered in the broadcast option. Our approach uses a register called the multicast register. This register is previously loaded with a communication pattern which identifies the receiving DSPs when the multicast option is used. The goal is to reduce the bandwidth used by the transmission and to allow all processors in the system to initialize a transfer simultaneously. A multicast system requires 1W + 1R access time for the whole transmission and allow up to n transfers to be conducted simultaneously by different processors.

To implement the system, we allocate 64 Kwords for interprocessor communication. This permits any processor to communicate with any other processor to form a fully connected topology. For each 2K address increment (corresponding to the size of the dual-port RAM module) in the first 62 Kwords in the interprocessor communication address region, only one processor is accessed at a time. Thus 31 DSPs and the MicroVAX can be accessed independently from any other processor. When the last 2K addresses in the interprocessor communication region are accessed with a "write", all processors corresponding to the pattern previously set in the multicast register of the sender. will be accessed simultaneously. This is accomplished with an address decoder, a 2:1 multiplexer. and the multicast register.

Address lines AB11-15 set the decoder outputs which are connected to the MUX. The signal DP generated by the dual-port RAM address comparator. enables the address decoder. The multicast register outputs are also connected to the MUX. If the last 2 Kwords in the interprocessor communication region are accessed, the 32nd output of the decoder is set to "0" and allows the pattern in the multicast register to be multiplexed. otherwise the outputs of the decoder are multiplexed assuming that DP is set. The MUX outputs provide the dual-port memory module select lines. Figure 4.1 shows the block diagram.

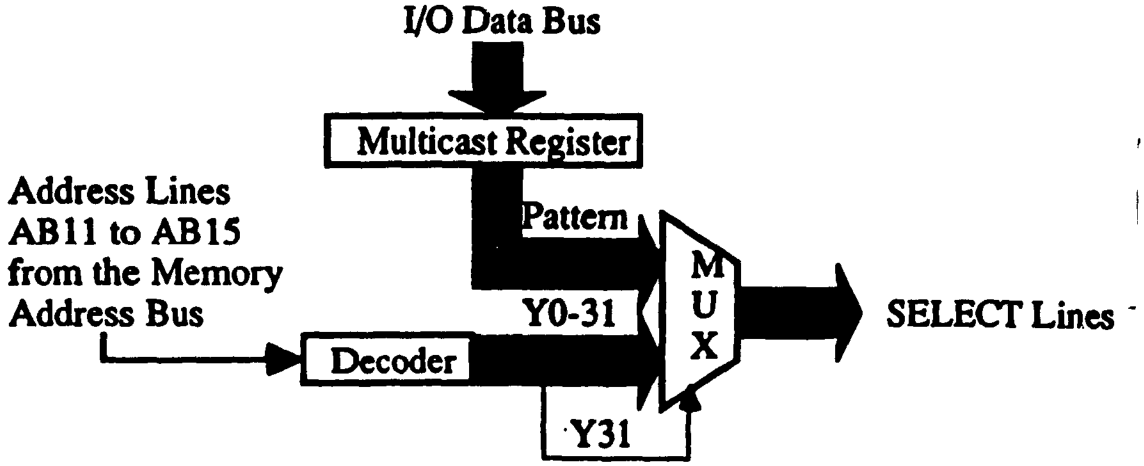


Figure 4.1 - Simplified block diagram of the multicast system.

The only drawback is that it requires as many select lines on the connection bus as there are processors in the system. Since 32 processors are present, we must allocate 32 select lines. This means that standard buses such as Multibus, Q-bus, VME-bus, etc..., cannot be considered in the internal bus structure.

4.2 Connection bus interface

All connection bus interface chips can drive up to 64 mA (I_{o1}). FCT devices [25] have been selected because they are very fast and have a CMOS internal structure surrounded by the bipolar technology. The rival technology to FCT is the BICMOS, but at present FCT is a bit faster. BICMOS is promising and will probably surpass the speed of FCT, but at present this is not the case. They can drive other FCT receivers and bring the voltage to CMOS level, thus lowering power consumption.

Signals	Descriptions
D(0-31)	Data
A(0-23)	Address
SELECT(0-31)	Dual-port RAM module selection lines
R/W	Read and write
STRB	Memory access strobe
READY	Indicates to the processor that data is available
BUSY	Indicates an access conflict in a dual-port module
INT	Interrupt for inter-processor communications
XF0	Interlock signal (output)
XF1	Interlock signal (input)
H1	Clock used as reference
H3	Clock used as reference (inverse polarity of H1)
VCC	+5 volts DC
GND	Ground

Table 2 - Signals for the connection bus.

The connection bus interface consists mainly of four FCT transceivers for the data path. three FCT drivers for the 24 address lines and other drivers for various control lines as well as receivers for status lines. Table 2 describes the different signals on the connection bus.

4.3 Wait-state generator

The wait-state generator is based on a 16R6 PAL [11]. It can generate up to two wait-states when the processor accesses the dual-port RAM modules. This is accomplished by sending a READY signal to the TMS320C30 at the appropriate time. The number of wait-states is selected by software. A BUSY signal generated by the dual-port RAMs indicates to the wait-state generator that it must extend the access cycle due to an address conflict with another processor. A maximum of 30 ns is required for the slaves to recognize the BUSY signal generated by the master (dual-port RAM modules have one master and three slaves). This extension when BUSY is considered, could add another wait-state. depending on the capacitive load on the connection bus.

The wait-state generator is synchronized with the clock signal H3 (inverse polarity of H1). It checks the DP signal from the dual-port RAM address comparator. the R/W line. as well as the BUSY signal.

CHAPTER FIVE

I/O SUBSYSTEM

For real-time control, a fast and efficient I/O subsystem is required. In this chapter, the design of such an I/O subsystem is presented. The primary task of this I/O interface is to link the processor to the external world. Most commercial I/O subsystems provide on-board A/D and D/A converters. The user is thus restricted to the capabilities of these devices.

The I/O subsystem proposed here, uses another approach. It incorporates a very flexible design which is able to interface with most A/D and D/A converters. Many external registers are provided to enable each I/O channel to be configured in various ways. The converters are located external to the I/O subsystem (preferably close to the input or output transducer). Each external converter (or other I/O device such as a counter) is linked to the I/O subsystem via a high speed serial line.

5.1 I/O subsystem requirements

The I/O subsystem communicates with the TMS320C30 via its secondary bus (I/O bus). The memory (primary) bus is not used for I/O since it is already used for memory accesses and interprocessor communication. Thus DMA transfers of large blocks of data on the primary bus can be performed without interruption by the I/O subsystem. The I/O interface should be as simple as possible. It should link with A/D and D/A converters serially and in digital form in order to minimize the number of wires ana to guarantee accurate data transmissions. Very accurate measurements in the order of 16-18 bits are predicted. A small distortion on an analog signal that propagates on a relatively long wire (coaxial or others) could decrease substantially the accuracy of the measurements or a signal sent to a D/A converter. Thus differential input with a good commcn mode rejection must be provided for each line receiver. The links must be fast since real-time controls are performed. The I/O subsystem must be flexible and adapt to 1 to 32-bit converters. Every channel should be configurable as input or output through software. Figure 5.1 shows the basic block diagram of the I/O subsystem for the serial channels.

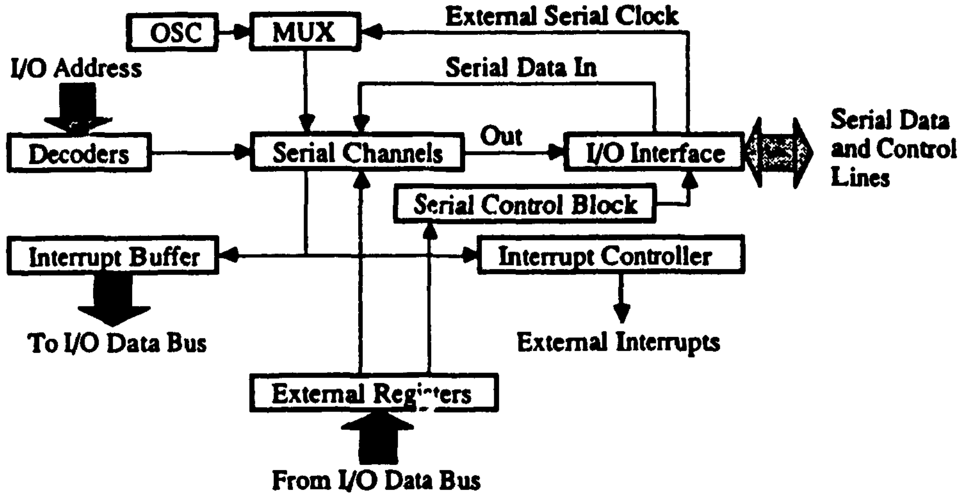


Figure 5.1 - I/O subsystem for the serial channels.

5.2 Serial-to-parallel and parallel-to-serial conversions

The 74FCT299 is a fast CMOS 8-input universal shift register. This shift register has 3-state outputs and can then be connected directly to the secondary bus without buffers, thus decreasing substantially the total chips count. Also, the parallel load inputs and flip-flop outputs are multiplexed to reduce the total number of package pins. This feature permits us to allocate through software. the channel as input or output. This IC can be cascaded easily such as four of them would support word lengths up to 32 bits. They are also available in SOIC which allows better board density. Figure 5.2 shows the implementation of the basic circuitry for the serial-to-parallel and the parallel-to-serial conversions.

Notice that serial-data-in are the serial data coming from an A/D converter with MSB first. and serial-data-out are the data sent to a D/A converter with MSB first. DSO and Q7 are the serial data inputs and serial outputs for a right shift respectively. OE1 and OE2 are output enable pins, and S0. S1 are mode select inputs. S0 and S1 allow to perform some basic operations such as a parallel load and a shift right. The left shift is not used since all external converters operate with the MSB first as a serial communication protocol. Thus each DS7 will be connected to the preceeding Q0 while the Q0 for the shift register interfaced with byte 0 and S7 for the shift register interfaced with byte 3 are left open and tied to ground respectively. MR. the asynchronous master reset input (active low) could be connected to the sytem reset to garantee a known initial state.

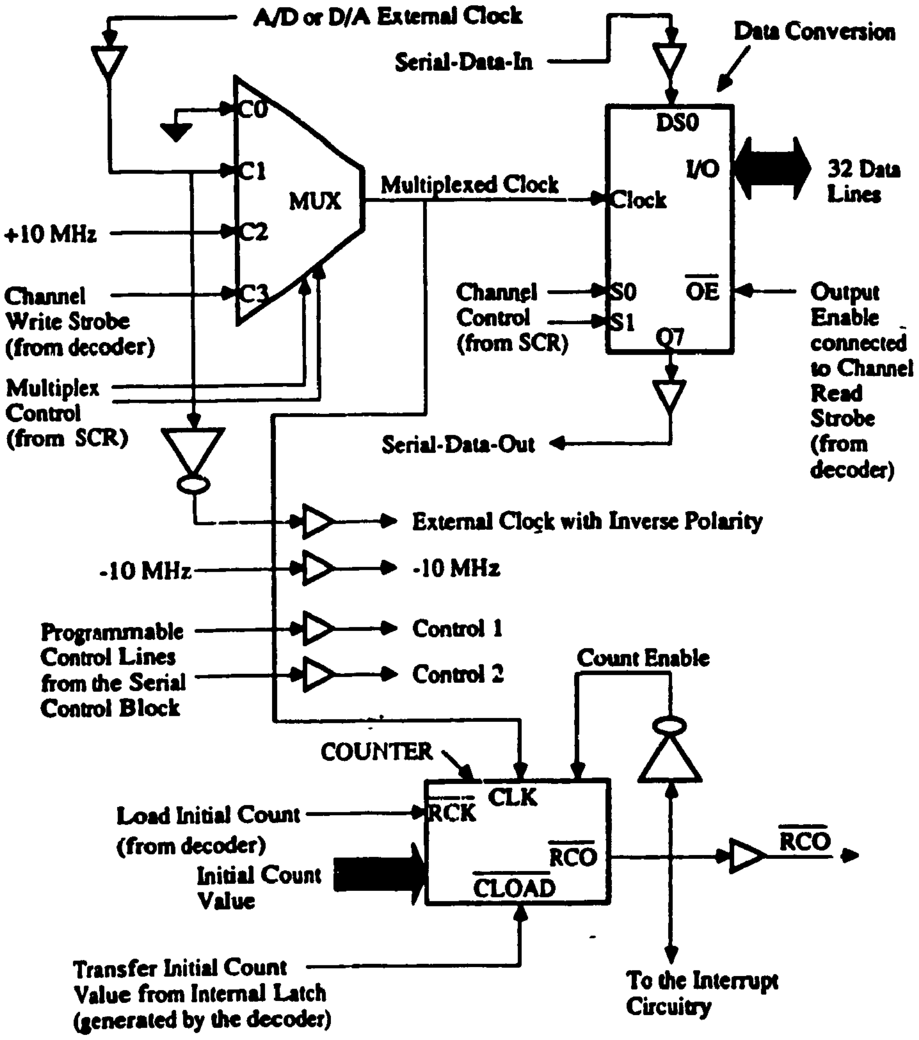

A/D or D/A External Clock
Serial-Data-In
Data Conversion
C0
C1
C2
C3
MUX
Multiplexed Clock
DSO
I/O
Clock
32 Data Lines
+10 MHz
Channel Write Strobe (from decoder)
Multiplex Control (from SCR)
Channel Control (from SCR)
S0
S1
Q7
OE
Output Enable connected to Channel Read Strobe (from decoder)
Serial-Data-Out
External Clock with Inverse Polarity
-10 MHz
-10 MHz
Programmable Control Lines from the Serial Control Block
Control 1
Control 2
Count Enable
COUNTER
CLK
RCK
Load Initial Count (from decoder)
Initial Count Value
RCO
CLOAD
RCO
Transfer Initial Count Value from Internal Latch (generated by the decoder)
To the Interrupt Circuitry

Figure 5.2 - Basic circuitry of one serial channel.

5.3 Number of serial channels

Figure 5.2 shows the basic circuitry for one input/output serial channel. Each I/O subsystem contains 16 such I/O channels.

5.4 Selection of the clock

When a serial channel is configured as input for data acquisition, the clock is provided by the A/D converter. This clock is sent with the serial data simultaneously. The rising edge of this clock is then used to enter the serial bit into the shift register and to shift right one bit. The clock in this case is not continuous and the number of rising edges corresponds to the word length transmitted. The maximum transmission rate is 10 Mbits/s over 12 meters which corresponds to the speed limit of the RS422 drivers and receivers.

The same basic approach applies when a serial channel is configured as output. In this case. the clock is provided on the I/O subsystem board when the maximum transmission speed can be considered, or sent by circuitry surrounding the D/A converter. when the transmission speed must be lower. A 20 MHz oscillator on the I/O subsystem board generates the required maximum clock. A D-type flip-flop is configured as a divide-by-two which results in two 10 MHz clocks of inverse polarities.

The 10 MHz clock is ideal since it is equal to the maximum transmission rate of the RS422 protocol. The +10 MHz can be used to drive the channels configured as output where the -10MHz is sent to the D/A converters. This configuration guarantees a pre-defined set-up time required by most D/A converters for the serial data before the rising edge of the clock to latch data properly.

This on-board clock generation allows the peripheral circuitry around the D/A converters to be minimized and simplified. This feature is desirable when converters are situated in very restricted spaces. But when the D/A converter does not accept input data stream at 10 MHz. then the required clock must be provided off the interface board. preferably in the D/A converter itself. The two internal timers in the TMS320C30 could have been used for this purpose for up to 8 MHz. but this solution restricts ourself to a maximum of two output channels. So, the external clock is sent to the I/O subsystem which drive the corresponding channel. An inverter provides the same clock shifted by 180 degrees that is sent back to the D/A converter. As for the 10 MHz configuration, a minimum set-up time is available.

A mechanism must be provided for each serial channel to select the proper clock. This is achieved with multiplexers. One 4:1 multiplexer is required for each serial channel. In order to improve the board density. eight 74AS153 dual 1 of 4 data selectors/multiplexers constitute the multiplex circuitry for all 16 channels.

50 ns separate the rising edges of the +10 MHz and the -10 MHz clocks. A maximum of 10 ns are required for the +10 MHz to propagate through the 4:1 multiplexer and another 10 ns maximum from CP to Q7, the clock input and serial data output of the shift registers respectively. If we assume that the propagation delays through the RS422 drivers and receivers are the same, and the wires have the same length and load capacitance. then a minimum of 30 ns is garanteed for the data latch set-up time in the D/A converter. For the external clock, the set-up time is reduced by the propagation of the inverter that generates the opposite polarity clock.

5.5 Serial channel controls

Parallel load. shift right and left, and hold are possible operations on each serial channel through the control lines S0 and S1. While S0 is high. S1 would determine whether a parallel load (1) or a shift right (0) is performed at the rising edge of the clock. Thus four control bits from two external registers named the serial control registers (SCR) are required to multiplex the correct clock and to control a serial channel.

A parallel load could be accomplished by selecting input C3 of the 4:1 multiplexer which is connected to one output of an address decoder which goes low when a "write" is done to the address corresponding to the serial channel. When the strobe goes back to high, the data are latched by the shift register if S0 and S1 were previously set to a high level.

We could receive serial data by selecting C1 (A=1, B=0) and setting S0 and S1 for a right shift. Changing A and B to "0" and "1" respectively, will multiplex the

+10 MHz clock through the input C2 and send data out at maximum rate. Other operations are possible such as holding the data or ignoring the external word. Notice also that four bits were assigned in order to simplify the circuitry and to provide independent control over the multiplexers and the shift registers.

5.5.1 Serial control registers

Two 32-bit serial control registers (SCR) are provided to control the 16 serial channels. Eight transparent or edge-triggered latches can be used to construct the SCRs. Some attention must be paid when an attempt is made to perform two operations in one cycle, especially when the channel is driven by a high speed clock. For example, if we select the hexadecimal command "6" to send data and multiplex the +10 MHz at once, some problems may occur. Because the +10 MHz is continuous and runs asynchronously from the DSP, and the four shift registers in the channel may represent different set-up times, a possibility exists that one shift is performed in a particular shift register and not in the others from the same channel. This results in false information being sent due to the difference in the set-up times. One way to correct the problem is to make sure that the rising edge of the +10 MHz clock does not reach the shift registers in less than 5.5 ns corresponding to the minimum set-up time from the serial data to the clock input of the 74FCT299. The 74AS153 has a minimum propagation delay from low to high of 2 ns. Thus 3.5 ns should be added to guarantee a correct operation. This value is for the worst case, for example the 74FCT299 has a typical set-up time of 0.5 ns which is far from the 5.5 ns required. With typical devices, both operations could be performed simultaneously. So, the user should be careful when attempting to perform two operations on the same channel simultaneously. A fixed delay element could be introduced in the channels to garantee proper timing. This solution imposes subtantial constraints in the synchronization circuit (not shown in Figure 5.2) to operate properly in the 0-10 MHz range. A first implementation of the system does not provide an internal clock generator and associated multiplexers. This simplifies the I/O subsystem at the expense of little more external circuitry.

Eight 74FCT574 constitute the SCRs. The output enable pins are connected to ground and the clock inputs to one of two select lines from an address decoder.

5.6 Counters

A counter must be provided for each serial channel to keep track of the number of bits received or transmitted. This counter is also used to transmit a flag indicating that a complete new word is available to be read. or to indicate that the transmission is completed.

The 74LS592 is an 8-bit binary counter with an input register. The eight inputs are connected to one byte of the secondary bus. Thus four counters per 16 channels would be connected to the same byte making four 32-bit on-chip counter registers. Each counter must be initialized by writing the word length. This is accomplished by tying RCK to one output of an address decoder. For every word read or transmitted. the counter must be loaded with the value in the internal register. This is done by connecting CLOAD to the output of an address decoder which goes low when a 'read' or a 'write' is performed on the respective channel. Thus. one dummy access must be performed to initialize the serial channel. The value being loaded corresponds to the hexadecimal value 'FF' minus the word length. At each rising edge of the multiplexed clock. the count increments. When it reachs "FF". a low-going RCO pulse is obtained. The CCLR input is connected to the system reset to permit all counters to start in a known state. Since the RCO is used as a status bit. it is necessary that RCO remains low between the completion of the transmission and the next corresponding serial channel access. When the serial channel is configured as output, the driving clock continues after the completion of the transmission and RCO remains low for only one clock period. To correct the problem, it is necessary to stop the counter by tying RCO to the clock enable through an inverter.

5.7 Polling and interrupt

A 16-bit external register named the status register is provided and accessible through the secondary bus. The status register indicates the state of the RCO pins for all 16 channels. This register is used when the polling option is preferred.

Each RCO (end-of-count) signal is sent to an interrupt controller. Because it is not always desirable to have all channels interrupt the processor, a maskable option is provided. This consists of an external 16-bit register named the mask register. The inputs of the mask register are connected to the secondary bus and each output goes to an OR-gate. The other input of the OR-gate is connected to RCO. The output of the OR-gate is connected to one input of an interrupt controller (active low). Writing a "1" to the mask register will mask the corresponding channel. Because the TMS320C30 has only four external interrupt pins. a 16-bit interrupt register is provided to indicate through the secondary bus. which channels wish to interrupt. Figure 5.3 shows the basic configuration.

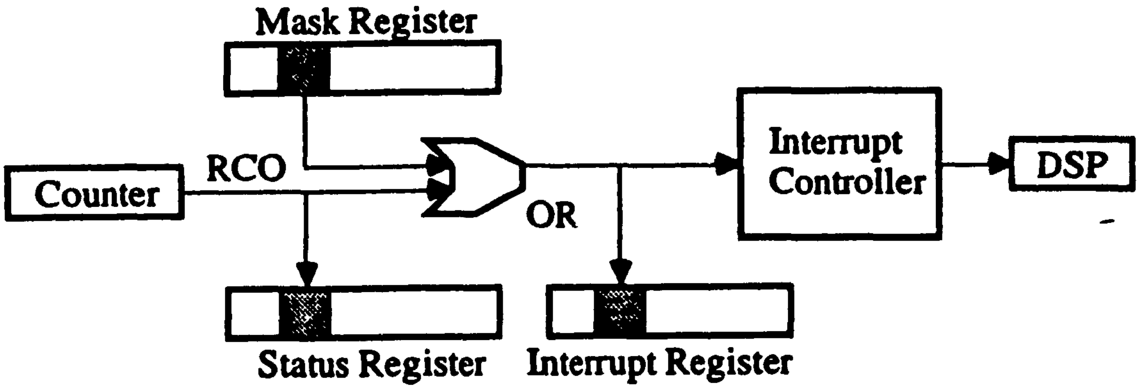


Figure 5.3 - Basic configuration for polling and interrupt.

5.7.1 Status and interrupt registers

Because the I/O channels run asynchronously from the processor, care must be taken that no RCO signals change state while the DSP is reading the status or the interrupt register. So. the content of both registers must be stable during 15 ns before the falling edge of H1 when MSTRB is low (active). H1 is an external 16.5 MHz clock generated by the TMS320C30. It is a good timing reference and provides the designer with tools to synchronize external circuits with the DSP. MSTRB is the memory secondary bus access strobe. The 15 ns correspond to the minimum data valid set-up time required by the TMS320C30 to guarantee a good reading. Thus two 74FCT574s would be required for each register. The output enable pins of each register are connected to a different output pin of an address decoder, while all input clock pins are tied together with the H1 clock. Thus both registers would be updated each 60 ns at the rising edge of H1. Notice that RCO goes back to "1" when the serial channel is accessed, and not the registers.

5.7.2 Mask register

The mask register consists of two 74FCT574s with the output enable pins connected to ground and the clock input pins connected to one output of an address decoder.

5.8 Serial control block

The serial control block provides the user with the ability to send pre-programmed signals to the external converters. Because the programmer can send various signals with specific polarities and pulse lengths, direct control over the converters can be performed.

Two configurable control signals are assigned to each serial channel. The serial control block consists basically of a 32-bit external register (polarity register), 32 8-bit counters with internal latch, and an XOR logic block as shown in Figure 5.4.

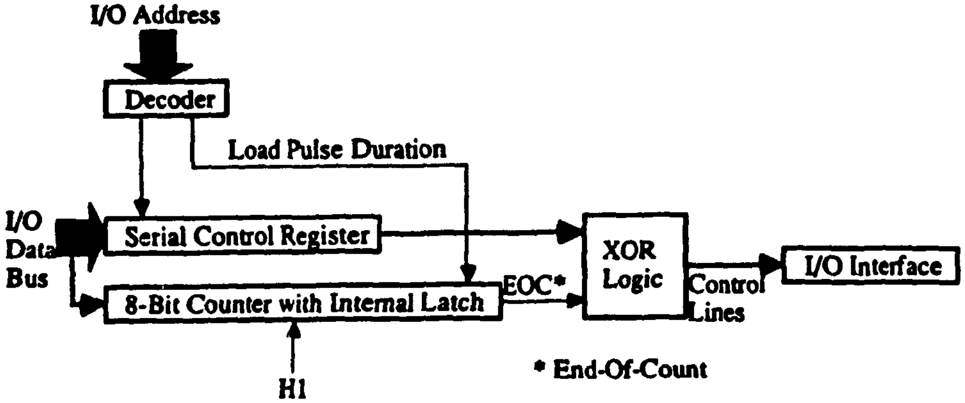


Figure 5.4 - The serial control block.

The polarity register consists of four 74FCT574s with the inputs connected to the secondary data bus and the outputs to the XOR logic block. The output enable pins are tied to ground and the clock inputs to one output of an address decoder.

32 74LS592s are necessary for the complete pulse length control circuitry. Each line can be controlled independently. These counters are synchronized by the TMS320C30. Output pins of an address decoder are used to load the internal registers (pulse length registers). An address is dedicated to transfer this value to the counter. By writing a specific pattern to this address. it becomes possible to send up to 32 control signals simultaneously. An inverter links each RCO with CCKEN since the reference clock is continuous. The reset pins are connected to the system reset while the RCO pins are connected to the XOR-gate block. The serial control block works as follows:

- If the counter is disabled (loaded to "FF" or already reached "FF"). RCO is low. By writing "0" in the polarity register. a "0" is obtained at the output of the XOR-gate or control line. By writing a "1", a "1" is obtained.

-If the user wishes to use the counter to send a control pulse with a pre-loaded polarity and pulse duration. then RCO is "1" during the count, and if the bit in the polarity register is "0", then a positive pulse of known duration is obtained. If a negative pulse is preferred, then the bit in the polarity register must be set to "1".

A pre-settable pulse duration between 0 and 15.3 ms with increments of 60 ns can be obtained.

5.9 Interrupt controller

An external interrupt must be held low for at least one H1/H3 cycle to be seen by the TMS320C30. This external interrupt signal must be held low for less than three cycles. otherwise more than one external interrupt may be seen if the interrupts are serviced quickly. The PAL-based interrupt controller guarantees that these conditions are met. Two 16R8 PALs [11] are configured the same way. Each of these PALs is programmed as an interrupt controller which is able to support 9 interrupt entries and generate two independent external interrupts compatible with the TMS320C30 interrupt specifications. The clock H3 running at 16.5 MHz is used as a time reference for both PALs. The first PAL accepts the interrupt flag from either the expansion board (DRAM error detected), or the I/O parallel bus (communication with the MicroVAX). and the eight first end-of-count flags from the serial channels. It then generates INT0. the highest priority interrupt when the expansion board or the parallel I/O bus (secondary bus) wishes to interrupt, or INT1. the second priority interrupt when at least one end-of-count flag in the first 8 serial channels is activated. The second PAL works exactly the same way. except that INT2 is generated when another processor wishes to communicate through the dual-port RAM modules. and INT3. the lowest external priority interrupt. when at least one end-of-count flag is set for the last 8 serial channels. Notice that the programmer can mask through software any of these external interrupts. Also notice that many configurations are possible. If only two channels are used. they should be split between the two interrupt controllers in order to avoid looking at the interrupt buffer.

5.10 I/O interface

The I/O interface consists of 16 serial input/output channels. two general purpose serial ports. and a parallel port. Because fewer transmission errors are introduced when the signals are sent in digital instead of analog form, serial channels carrying digital information have been adopted in all cases between external converters and the parallel computer. Coaxial cables could have been used. but because of the number of wires involved and the size of these wires. twisted pair wires have been chosen. The RS-422 protocol is a good choice. It offers up to 10 Mbits/s of transmission speed over 12 meters. and uses differential inputs for better performance. Thus serial channels for input data. output data. input serial clocks. output serial clocks. and serial output channels for the control lines. supported by the EIA standard RS422A drivers and receivers [31] constitute the entire serial input and output channels of the I/O interface.

Two serial ports are provided by the TMS320C30.·To allow more flexibility, all lines in these ports are available on two general purpose connectors. The same idea applies to the I/O parallel port. In order to minimize the chip count on the processor board, and not to stick to a particular bus protocol, all 32 I/O data lines, 13 I/O address lines, and a few control lines, are linked directly to a parallel connector for future expansion.

5.11 Decoders

The DSP accesses different parts of the system by reading or writing to an external bus. The address associated with the external access is decoded by address comparators and address decoders. It results an active signal which is used to enable the proper part of the system. The decoders (address comparators and address decoders) accept all address lines, the R/W line, and the access strobe as inputs. The access strobe confirms the validity of the address lines and the decoders interpret it as a valid external access. The R/W line is used to decode write-only or read-only external registers.

CHAPTER SIX

DUAL-PORT RAM MODULES

One of the biggest problems in novel computer architectures as been the implementation of an efficient mechanism for interprocessor communication. Recent innovations in dual-port RAM designs have effectively solved this problem.

The communication module was developed to optimize fully a new interprocessor communication link integrated in our supercomputer architecture. 496 of such modules would permit 32 digital signal processors (DSP) to be implemented in a fully connected topology.

Up to 31 modules would be connected to one of the 32 connection buses. It is thus important to include on each module an interface able to drive a high capacitive load. But the module should be more intelligent to optimize fully the throughput of the computer. The module must assist interprocessor signalling by providing an interrupt facility on both ports. A sender identification scheme must also be implemented (to avoid the receiving processor reading up to 31 modules) to access the right module that generated the interrupt. Finally, this configuration reduces significantly the access conflicts to memory, but must be able to deal with conflicts to the same address location by providing a fast access arbitration logic.

The following describes the hardware of such a module. Emphasis is performed on the design. The interface is compatible with special new techniques developed for the new supercomputer such as the multicast system which permits one processor to write to 31 modules simultaneously, and a multiplexed interrupt identification scheme which allows a processor to read through the 32-bit data bus, all interrupt lines from the modules. It also provides an example of an implementation of the new dual-port RAM devices in an intelligent interprocessor communication scheme with a performance superior to conventional parallel architectures. Each module provides 2Kx32 bits of buffering area with special memory addresses which when written, generate an interrupt on the opposite port which is normally used for interprocessor signalling. This signalling feature provides a useful tool for the synchronization of multiple processors in a true multitasking environment.

CHAPTER SIX - DUAL-PORT RAM MODULES

6.1 Dual-port module interface

The dual-port module (DPM) consists basically of one dual-port RAM set (one master and three slaves), and one dual-port module interface (DPMI) on each port. The main tasks of the DPMI are to increase the driving capability of the dual-port RAMs during a DSP "read" for the connection data bus, to decrease the maximum capacitive load on the connection bus for the eleven least significant address lines and the R/W line, and to provide a mechanism to multiplex the interrupt signal to an assigned position onto the communication data bus in order to support an interrupt identification scheme (IIS).

6.1.1 Dual-port data bus interface

The dual-port data bus interface is built using four 74FCT645A tranceivers on each port. An internal R/W line is connected to the T/R (transmit/receive) pins of the transceivers. The output enable pins are tied to the left or right chip enable pins of each dual-port RAM and connected to one of the 32 select lines sent by the multicast system using a jumper. Bus A of the tranceiver is connected to the dual-port RAMs and bus B to the connection bus.

6.1.2 Receivers

Receivers are used to minimize the capacitive load on the communication bus and to improve the overall performance. This reduces by at least four times the total capacitive load on the bus. Receivers are necessary for the eleven address lines (AB0-10) and the R/W line. Notice that the buffered R/W line is referred as the internal R/W line used to control the transceivers and the dual-port RAMs.

Three receivers of type 74FCT244A are required per DPM. The output enable pins may be tied to ground.

6.1.3 Interrupt identification scheme

The interrupt identification scheme (IIS) permits the processor to identify in

CHAPTER SIX

DUAL-PORT RAM MODULES

One of the biggest problems in novel computer architectures as been the implementation of an efficient mechanism for interprocessor communication. Recent innovations in dual-port RAM designs have effectively solved this problem.

The communication module was developed to optimize fully a new interprocessor communication link integrated in our supercomputer architecture. 496 of such modules would permit 32 digital signal processors (DSP) to be implemented in a fully connected topology.

Up to 31 modules would be connected to one of the 32 connection buses. It is thus important to include on each module an interface able to drive a high capacitive load. But the module should be more intelligent to optimize fully the throughput of the computer. The module must assist interprocessor signalling by providing an interrupt facility on both ports. A sender identification scheme must also be implemented (to avoid the receiving processor reading up to 31 modules) to access the right module that generated the interrupt. Finally, this configuration reduces significantly the access conflicts to memory, but must be able to deal with conflicts to the same address location by providing a fast access arbitration logic.

The following describes the hardware of such a module. Emphasis is performed on the design. The interface is compatible with special new techniques developed for the new supercomputer such as the multicast system which permits one processor to write to 31 modules simultaneously, and a multiplexed interrupt identification scheme which allows a processor to read through the 32-bit data bus, all interrupt lines from the modules. It also provides an example of an implementation of the new dual-port RAM devices in an intelligent interprocessor communication scheme with a performance superior to conventional parallel architectures. Each module provides 2Kx32 bits of buffering area with special memory addresses which when written, generate an interrupt on the opposite port which is normally used for interprocessor signalling. This signalling feature provides a useful tool for the synchronization of multiple processors in a true multitasking environment.

CHAPTER SIX - DUAL-PORT RAM MODULES

6.1 Dual-port module interface

The dual-port module (DPM) consists basically of one dual-port RAM set (one master and three slaves), and one dual-port module interface (DPMI) on each port. The main tasks of the DPMI are to increase the driving capability of the dual-port RAMs during a DSP "read" for the connection data bus, to decrease the maximum capacitive load on the connection bus for the eleven least significant address lines and the R/W line, and to provide a mechanism to multiplex the interrupt signal to an assigned position onto the communication data bus in order to support an interrupt identification scheme (IIS).

6.1.1 Dual-port data bus interface

The dual-port data bus interface is built using four 74FCT645A tranceivers on each port. An internal R/W line is connected to the T/R (transmit/receive) pins of the transceivers. The output enable pins are tied to the left or right chip enable pins of each dual-port RAM and connected to one of the 32 select lines sent by the multicast system using a jumper. Bus A of the tranceiver is connected to the dual-port RAMs and bus B to the connection bus.

6.1.2 Receivers

Receivers are used to minimize the capacitive load on the communication bus and to improve the overall performance. This reduces by at least four times the total capacitive load on the bus. Receivers are necessary for the eleven address lines (AB0-10) and the R/W line. Notice that the buffered R/W line is referred as the internal R/W line used to control the transceivers and the dual-port RAMs.

Three receivers of type 74FCT244A are required per DPM. The output enable pins may be tied to ground.

6.1.3 Interrupt identification scheme

The interrupt identification scheme (IIS) permits the processor to identify in one "read" access. which DSPs have sent an interrupt for interprocessor signalling. This is accomplished by tying each interrupt pin of each DPM to a dedicated data line onto the connection bus through a driver which is enabled when the IIS option is used. When a "read" is performed in the multicast region (last 2K addresses in the interprocessor communication region). a select line (INTSEL) is set to "0" which enables the drivers and multiplexes all interrupt lines (one port only) to the data bus. This feature also minimizes the connection bus width resulting in a compact system. The driver is of type 74FCT244A. Only one 74FCT244A is required for the two ports.

Because all interrupt lines on the same port of all DPMs connected to the same communication bus. must be tied together and forwarded to an interrupt controller situated on the DSP board. an open-collector driver must be provided on each port of each DPM to isolate the IIS from the interrupt line on the connection bus. One 74ALS09 quadruple 2-input positive-AND gates with open-collector outputs is provided per DPM for this purpose.

6.2 Dual-port RAM

The dual-port RAM is expanded from 8 to 32 bits by using a master/slaves configuration. This resolves the busy lock-up problem where only one dual-port RAM does the arbitration. The BUSY signal generated by the master is low when both ports are accessed at the same address simultaneously. It is sent to the three slaves and to a wait-state generator on the DSP board. The BUSY line is open-drain which allows the designer to connect all BUSY lines of all DPMs on the connection bus together. This configuration is preferred to 31 separate lines which would considerably expand the size of the connection bus. A 330 ohms pull-up resistor is required. With the 35 ns access time version. a maximum of 30 ns is required for the master to set the busy line and the slaves to recognize it after the beginning of the dual-port RAM access. The CY7C136 and CY7C146 from Cypress or the IDT71321 and IDT71421 from IDT (Integrated Device Technology) show the same characteristics.

Because the data pins of these dual-port RAMs are isolated from the connection bus by transceivers. the output enable pins can be tied to ground. The write enable pins are connected to the internal R/W line on the proper bus. Figure 6.1 shows the block diagram of one dual-port module. The module is compatible with any 32-bit processor. The module interfaces only two processors. This scheme results in more interface logic but adds a significant flexibility in building complex computer architectures.

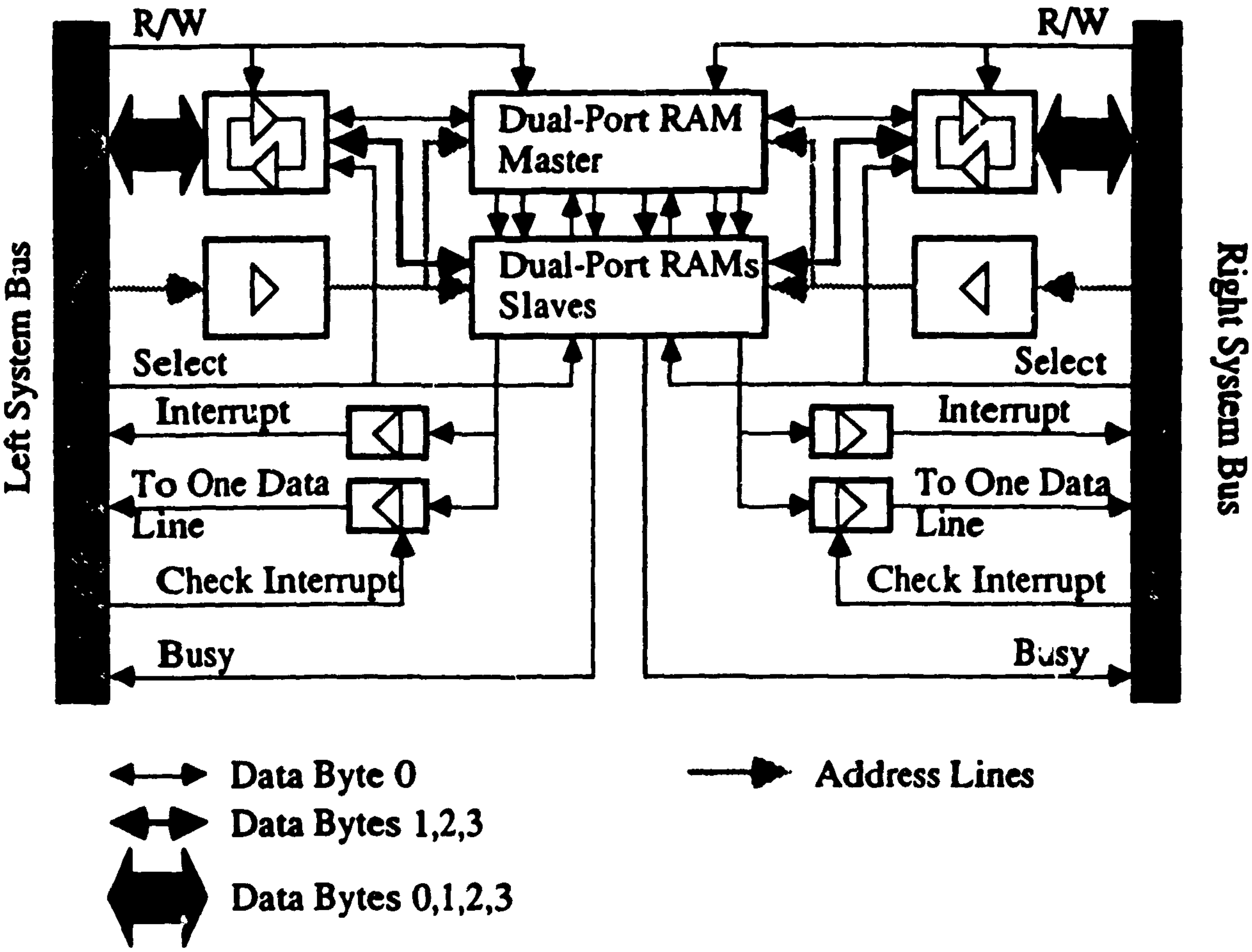


Figure 6.1 - Block diagram of one dual-port module.

The next important design decision is the access time required by the dual-port RAMs. The multicast system requires a maximum of 23.5 ns to select the DPMs. During this period, the eleven least significant address bits propagate through 74FCT244A drivers on the DSP board and through the 74FCT244As used as receivers on the DPMs. Since up to 31 DPMs are connected to the same connection bus, a maximum of 310 pF (excluding the bus line capacitance and the input capacitance of the interface of the expansion board) could be present at each output of the address drivers. The maximum propagation delay of the 74FCT244A is 4.3 ns for a capacitance of 50 pF. It could increase to a maximum value of 12.1 ns for a fully populated bus (excluding the expansion board). Thus the address could reach the dual-port RAMs in 16.4 ns. Because the propagation through the multicast system is larger, it must therefore be considered instead in our calculations. Since the TMS320C30 requires a maximum of 35 ns for a zero wait-state memory access, at least one wait-state must be considered for a "read" to the DPMs. When the data are valid from the dual-port RAMs, it must propagate through the 74FCT645A transceivers on the DPM and through another set of transceivers on the DSP board. The 74FCT645A has a maximum propagation delay of 4.6 ns. For a processor "read", the DPM transceivers must drive the relatively high capacitive load of the communication data bus which could be as high as 372 pF (12 pF x 31 modules). The maximum propagation delay in this case could increase to 14.2 ns. This value added to the DSP transceivers propagation delays results in a value of 18.8 ns which must be considered for data to propagate from the DPM to the DSP. The total access time assuming a 0 ns access time for the dual-port RAM, is 42.3 ns for the worst case. As this value is already greater than 35 ns, we conclude that at least one wait-state must be generated when a "read" is done on a DPM. With one wait-state, the maximum access time required by the TMS320C30 increases to 95 ns. Thus the dual-port RAMs should have a maximum access time of 52.7 ns (95-42.3). But if we consider the propagation delay of the signals through the connection bus itself which has a length of 0.6 m, allowing a maximum of 6.6 ns/m, results in a one-way propagation delay of 4 ns. For a "read", 8 ns should be subtracted from 52.7 ns. This results in a dual-port RAM maximum access time required of 44.7 ns. The 45 ns access time version should work properly, but because some parameters such as the bus capacitance and the interface of the expansion board have not been considered, dual-port RAMs with a 35 ns access time should be selected in order to provide a safety margin and guarantee correct operations for all conditions.

6.3 Busy lock-up

To form a module. four 2Kx8 bit dual-port RAMs are necessary. This expansion in width implies that several dual-port RAMs in the module can be active simultaneously. Thus. on rare occasions. it is possible for one RAM arbitrator to activate the left BUSY and the other RAM arbitrator to activate its right BUSY resulting in a lock-up problem where both DSPs will wait indefinitely for their port to become free. The solution is to use the arbitration logic in one RAM called the master and to force the other RAMs called the slaves to follow it. The problem is that a maximum of 30 ns is necessary for the master to set the BUSY signal and the slaves to recognize it. Thus. if a risk of address conflict is present. a delay of 30 ns must be added to the access time to prevent reading a false value. This is the reason that a wait-state generator on the DSP board is provided.

Even when the problem is resolved for one module. it persists when more than one module is accessed at the same time such as when the multicast option is used. In this case. we have as many masters as there are receiving processors (up to 31). Since we cannot modify the arrangement of each module. a "watch dog" timer could be used to prevent processors waiting too long in a lock-up situation. When multicast with BUSY is performed. an internal timer in the TMS320C30 is set to an arbitrary value. If the counter has not been reset at the end of the multicast transfer. an internal interrupt is generated indicating a lock-up situation. This solution results in minimum external logic since circuitry is still necessary to set the READY line. Our approach is different. The wait-state generator works with a special mode that does not consider the BUSY from the dual-port modules. Thus no additional wait-states due to the contention problem will occur during a multicast transfer. When a bit dedicated for this special mode is changed. a flip-flop with BUSY connected to the clock input is enabled. If BUSY changed state during the multicast transfers. the output of the flip-flop will change value. Reading this bit through the 32nd data line when the IIS is accessed. indicates if the multicast transfers were successful.

CHAPTER SEVEN

INTERFACE BOARD

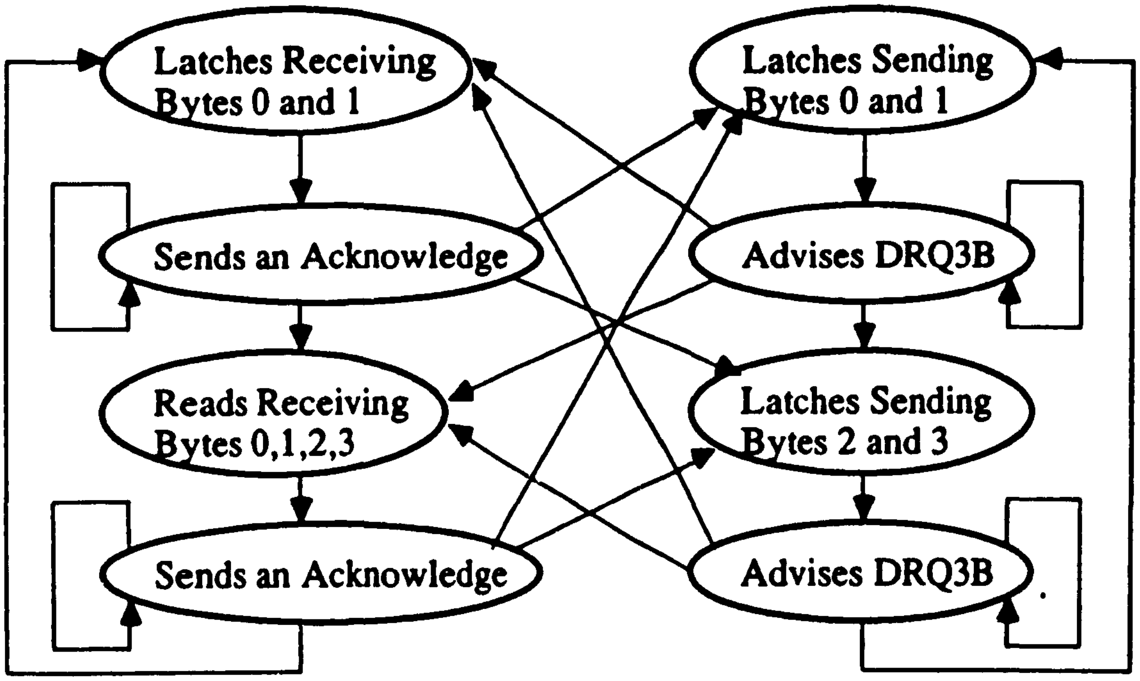

Latches Receiving Bytes 0 and 1
Latches Sending Bytes 0 and 1
Sends an Acknowledge
Advises DRQ3B
Reads Receiving Bytes 0,1,2,3
Latches Sending Bytes 2 and 3
Sends an Acknowledge
Advises DRQ3B

Figure 7.1 - State diagram for the interface board.

The interface board links the MicroVAX to the parallel computer. This interface board is connected to a connection bus with dual-port RAM modules in the same manner as the DSP boards. This results in a symmetrical system and provides the same interprocessor communication features encountered on the DSP boards. Thus. the MicroVAX is seen by the system as the 32nd processor. Because the MicroVAX is not intended to accompany the DSPs in real-time computation. but rather to provide a good interface to the user. the task of the interface board is mainly to perform transfer of code and data to and from the parallel computer. to generate system resets. as well as allowing program execution. The main objective is: a) to match the 32-bit system bus with the two 16-bit buses provided by the DRQ3B parallel interface board [33] in the MicroVAX; and b) to act fast enough in order to take advantage of the two buses configuration of the DRQ3B;

and c) to perform DMA transfers by using the same options as used by the DSP boards (such as the multicast, access arbitrations, and interprocessor signalling); and d) to download from the MicroVAX, an initialization program and loader in the external local memory of the DSP assigned as the communication processor before the parallel computer starts execution. The easiest way to address these requirements as well as providing all the options, is to use one of the DSP boards for the interface with the MicroVAX. The DSP board has the advantage of being fast and the board is already designed. The state diagram for the tranfers is shown in Figure 7.1. In order to perform the state diagram, a small interface board would be provided. Its block diagram is shown in Figure 7.2. The FUNCT pins are used to reset different parts of the parallel computer, and provide status bits.

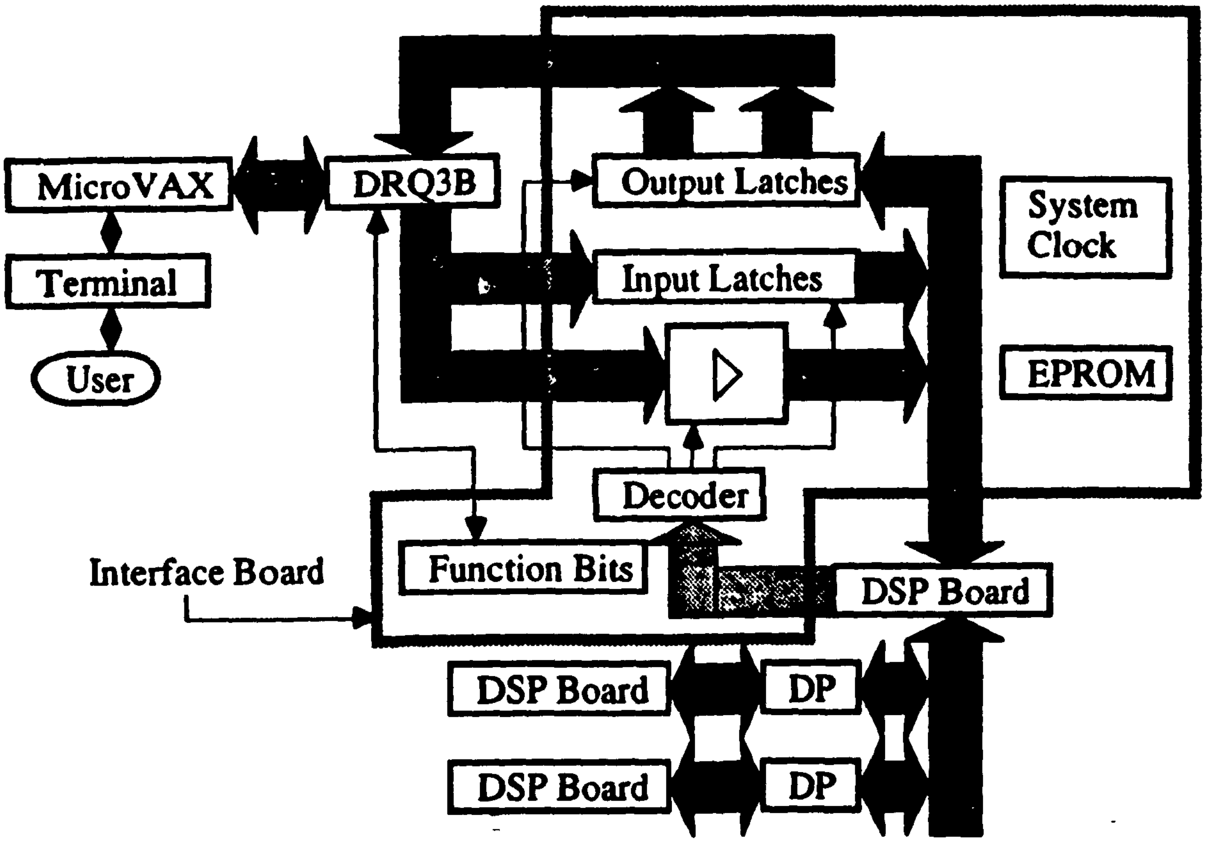


Figure 7.2 - Block diagram of the interface board.

CHAPTER EIGHT

CONCLUSIONS

Parallel processing has a bright future. The choice of the processor in a parallel system remains a key decision which must be made with respect to the needs of potential users. According to [34], although a variety of connection schemes exist. two fundamental types have emerged; the bus-based machines with a global memory. and the cube-based machines with typically "distributed" memory schemes that place local memory at each computing node. The last scheme resembles our approach. However our architecture is unusual in that it allows a direct communication path with all other nodes in the system.

The development of a new parallel computer architecture allows us to optimize the design for our immediate needs. It is possible to built a sophisticated parallel computer with only "off-the-shelf" components. Since the DRAM part of our design is subject to change. mainly because it is not a part of the basic system. its design was performed in order to identify more accurately the required signals for the connection bus. since DRAMs could be implemented on the expansion board.

A parallel computer designed for extensive real-time applications. must have a sophisticated I/O interface. Since this computer system is intended for research environments where external devices such as A/Ds and D/As are subject to change. the I/O interface must provide the ability to adapt to these changes under software control.

For high-performance memory. there is a trade-off between speed and chip count. The maximum capacitance must be split by increasing the size of the memory interface in order to optimize the throughput of the board. The number of levels in the decoding scheme must be kept low, in the order of one for a true zero wait-state memory board with a 35 ns maximum access time. RAMs with a common I/O should be selected to minimize the capacitive load. The decoding scheme adopted is flexible and very fast but restricts the number of memory banks to eight since a larger memory array would require the addition of drivers and additional propagation delays due to the increase of the capacitance between the processor and the additional comparators. One way to overcome this problem is to choose faster SRAMs at the expense of board density.

With the availability of dual-port static RAMs with internal arbitration logic and interrupt facilities, it becomes relatively easy to design a dual-port communication module well-suited for the present and future advanced computer systems. With the addition of a proper interface, the power of these dual-port RAMs can be significantly increased resulting in an improvement of the overall performance of the machine. This is accomplished by providing mechanisms that free the processors in repetitive and strategic tasks related to interprocessor communication. synchronization. and signalling.

In order to maintain a high degree of flexibility, many external registers were implemented. We avoided the inflexibility imposed by ROMs and instead use preload circuitry on the interface board for program initialization. This circuit permits us to download the initialization routines directly into one bank of the local memory of the DSP assigned as the communication processor while it is reset.

Future work on the parallel computer will include construction of a very simple operating system.

APPENDICES

APPENDIX A

DSP BOARD

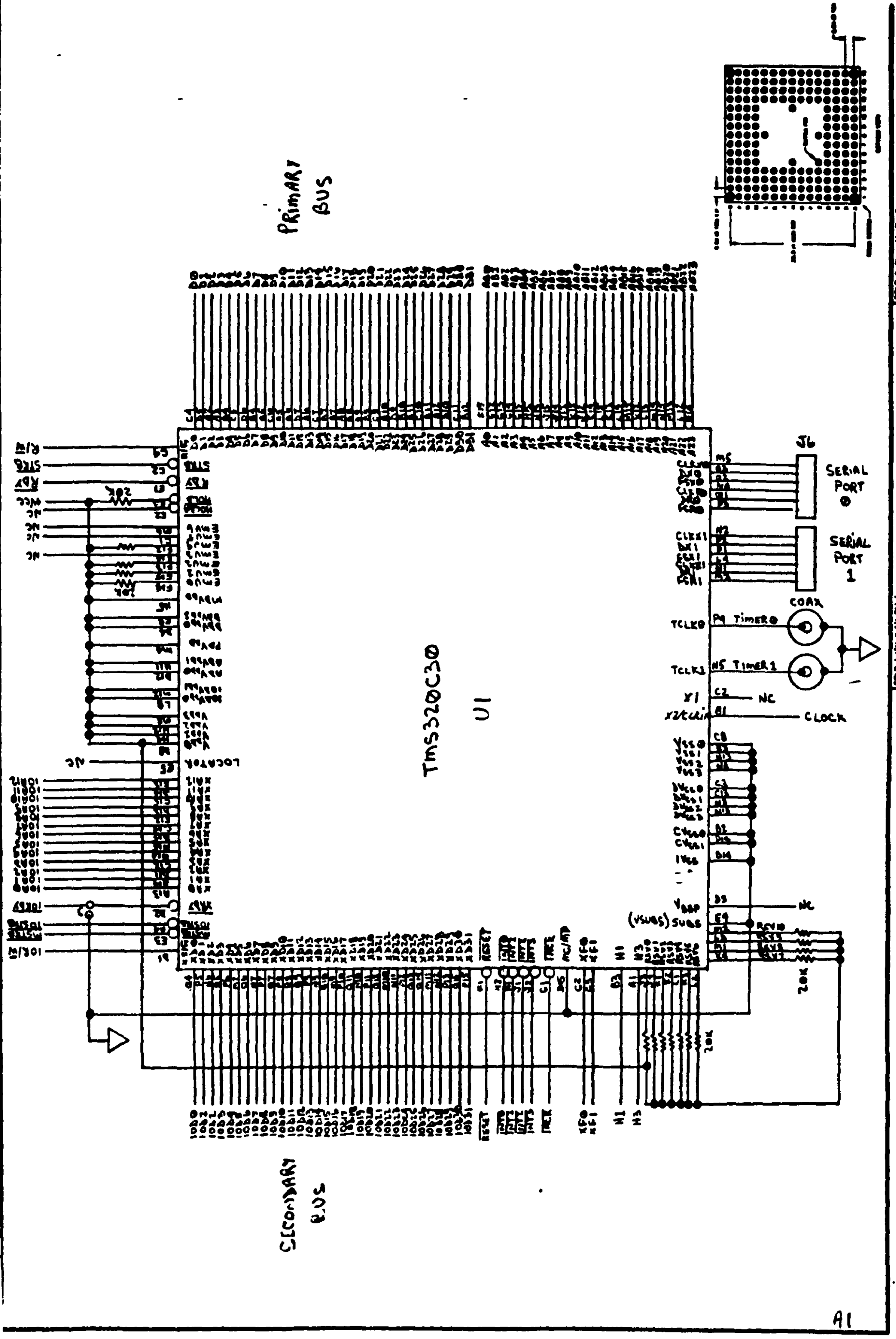
Sheet A1 - Digital signal processor (TMS320C30)
Sheet A2 - External registers decode
Sheet A3 - Connection bus interface
Sheet A4 - Multicast system (multicast register and decode section)
Sheet A5 - Multicast system (multiplexers)
Sheet A6 - Wait-state generator and interrupt controller
Sheet A7 - PAL equations for the wait-state generator
Sheet A8 - Simulation for the wait-state generator
Sheet A9 - Fuse plot for the wait-state generator
Sheet A10 - Fuse plot for the wait-state generator
Sheet A11 - JEDEC file for the wait-state generator
Sheet A12 - Simulation selective trace and history listing for the ws generator
Sheet A13 - PAL equations for the interrupt controller
Sheet A14 - Simulation for the interrupt controller
Sheet A15 - Fuse plot for the interrupt controller
Sheet A16 - Fuse plot for the interrupt controller
Sheet A17 - JEDEC file for the interrupt controller
Sheet A18 - Simulation selective trace listing for the interrupt controller
Sheet A19 - Simulation history listing for the interrupt controller
Sheet A20 - Serial control, configuration, wait state, and control registers
Sheet A21 - Allocation register, buffers, and receivers
Sheet A22 - Start-up circuitry

DSP board

Type	Description	Quantity
TMS320C30	Digital signal processor	1
74FCT138A	Fast CMOS 1-of-8 decoder	10
74FCT521B	Fast CMOS 8-bit identity comparator	2
74FCT645A	Fast CMOS non-inverting buffer transceiver	12
74FCT244A	Fast CMOS octal buffer/line driver	7
74F00	Quadruple 2-input positive-NAND gates	1
74FCT240A	Fast CMOS octal buffer/line driver	1
74FCT574A	Fast CMOS octal D register (3-state)	10
74FCT139A	Fast CMOS dual 1-of-4 decoder	1
74F257	Quadruple 2 to 1-line data selectors/multiplexers	8
16R6	PAL	1
74F32	Quadruple 2-input positive-OR gates	1
74F08	Quadruple 2-input positive-AND gates	1
74F21	Dual 4-input positive-AND gates	1
74F30	8-input positive-NAND gate	2
16R8	PAL	2
74FCT273A	Fast CMOS D flip-flop with clear	5
74FCT240	Fast CMOS octal buffer/line driver	2
74FCT827B	High-performance CMOS buffers	1
74FCT244	Fast CMOS octal buffer/line driver	3
74LS74	Dual D-type positive-edge-triggered flip-flops	1

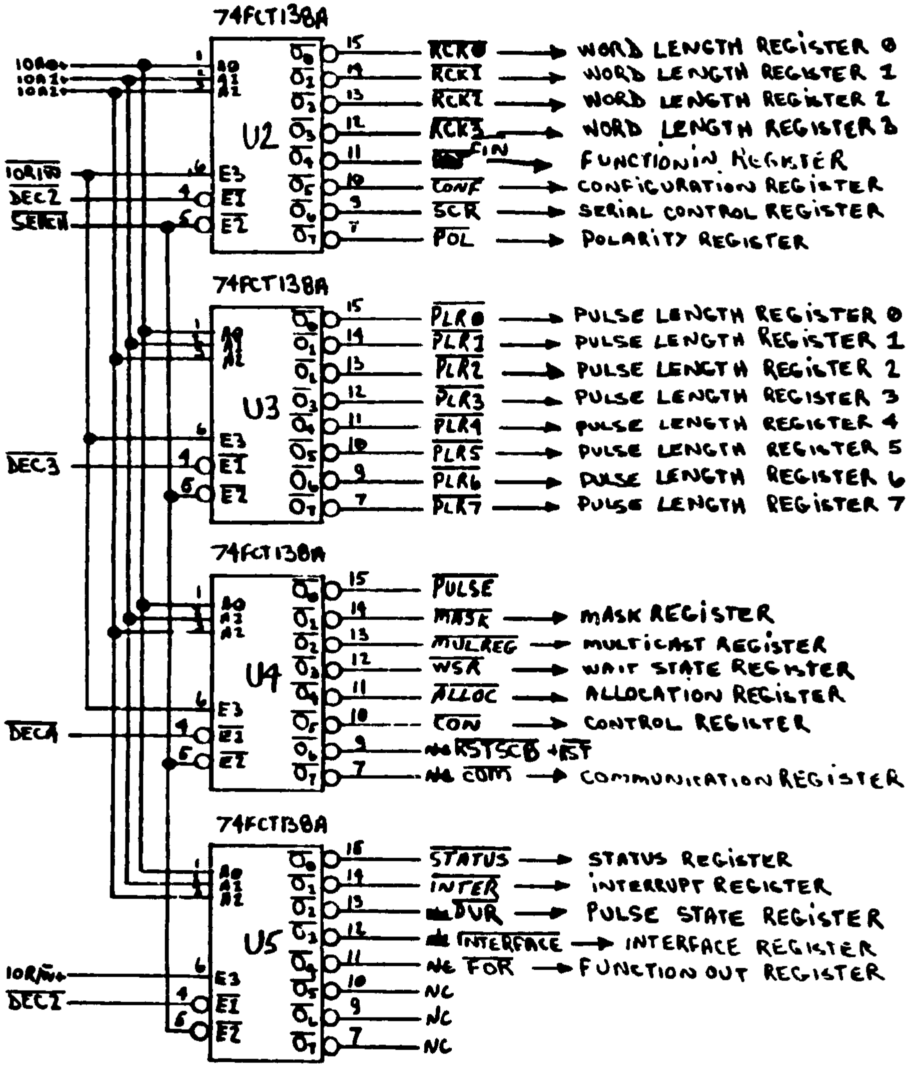
Total 73

DIGITAL SIGNAL PROCESSOR



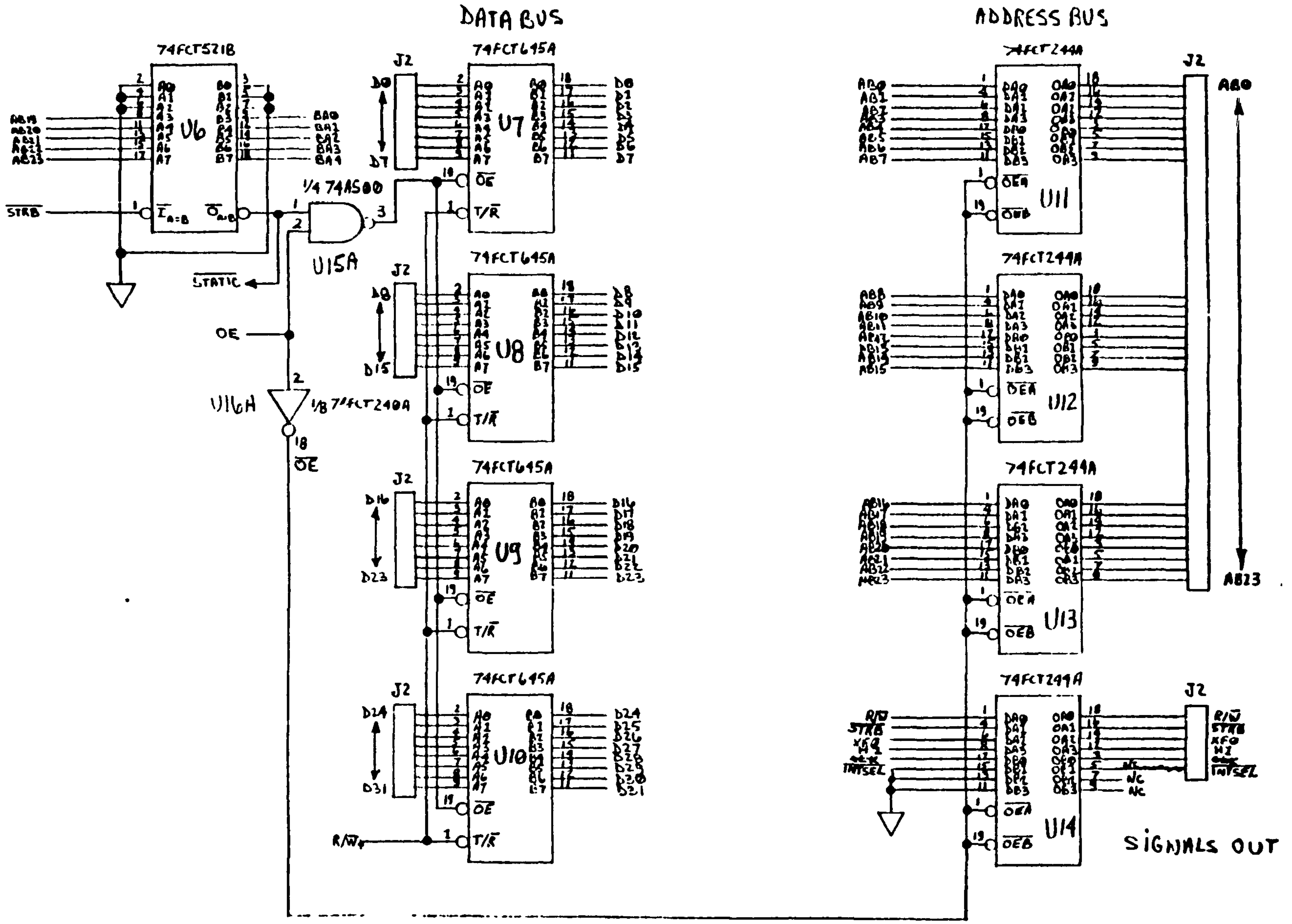
A1

PREPARED BY – ÉTABLI PAR	DATE	PAGE OF DE	PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME REGISTERS DECODE		IDENTIFICATION	



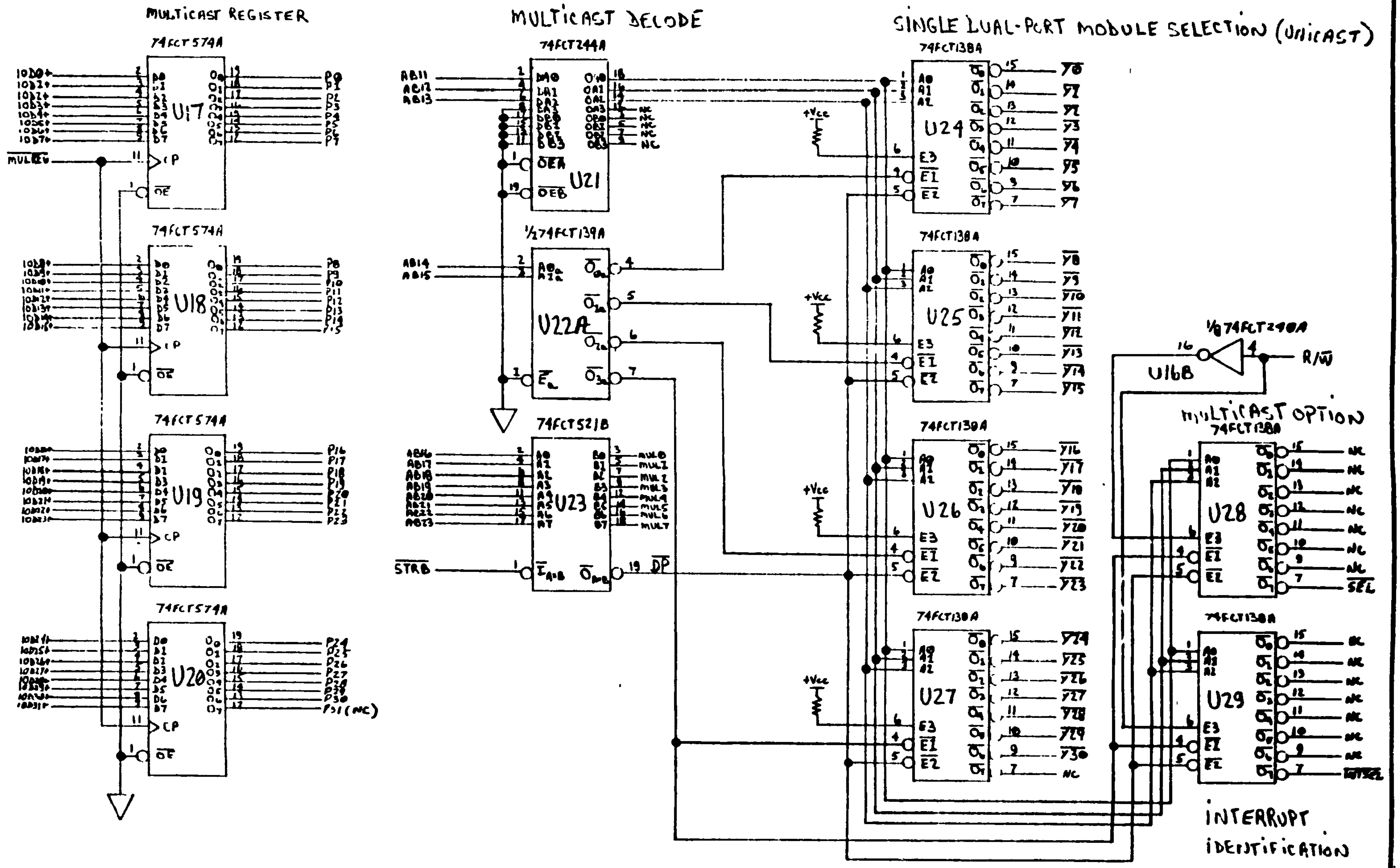
74FCT138A
U2
IORA0
IORA1
IORA2
IORI/W
DEC2
SERIN
A0 A1 A2 E3 E1 E2
1 2 3 6 4 5
O0 O1 O2 O3 O4 O5 O6 O7
15 14 13 12 11 10 9 7
RCR0 → WORD LENGTH REGISTER 0
RCR1 → WORD LENGTH REGISTER 1
RCR2 → WORD LENGTH REGISTER 2
RCR3 → WORD LENGTH REGISTER 3
FIN → FUNCTIONIN REGISTER
CONF → CONFIGURATION REGISTER
SCR → SERIAL CONTROL REGISTER
POL → POLARITY REGISTER
74FCT138A
U3
DEC3
PLR0 → PULSE LENGTH REGISTER 0
PLR1 → PULSE LENGTH REGISTER 1
PLR2 → PULSE LENGTH REGISTER 2
PLR3 → PULSE LENGTH REGISTER 3
PLR4 → PULSE LENGTH REGISTER 4
PLR5 → PULSE LENGTH REGISTER 5
PLR6 → PULSE LENGTH REGISTER 6
PLR7 → PULSE LENGTH REGISTER 7
74FCT138A
U4
DEC4
PULSE
MASK → MASK REGISTER
MULREG → MULTICAST REGISTER
WSR → WAIT STATE REGISTER
ALLOC → ALLOCATION REGISTER
CON → CONTROL REGISTER
RSTSCB → RST
COM → COMMUNICATION REGISTER
74FCT138A
U5
IORA/W
DEC2
STATUS → STATUS REGISTER
INTER → INTERRUPT REGISTER
DUR → PULSE STATE REGISTER
INTERFACE → INTERFACE REGISTER
FOR → FUNCTION OUT REGISTER
NC
NC
NC

PREPARED BY – ÉTABLI PAR	DATE	PAGE 1 OF	PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME CONNECTION BUS INTERFACE	IDENTIFICATION		



A3

PREPARED BY – ÉTABLIE PAR	DATE	PAGE 1 OF	PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME MULTICAST SYSTEM		IDENTIFICATION	



MULTICAST REGISTER
74FCT574A
U17
U18
U19
U20
CP
OE
MULREG
P0 P1 P2 P3 P4 P5 P6 P7
P8 P9 P10 P11 P12 P13 P14 P15
P16 P17 P18 P19 P20 P21 P22 P23
P24 P25 P26 P27 P28 P29 P30 P31 (NC)
MULTICAST DECODE
74FCT244A
U21
OEA
OEB
AB11
AB12
AB13
NC
1/2 74FCT139A
U22A
AB14
AB15
74FCT521B
U23
STRB
DP
MUL0 MUL1 MUL2 MUL3 MUL4 MUL5 MUL6 MUL7
AB16 AB17 AB18 AB19 AB20 AB21 AB22 AB23
SINGLE DUAL-PORT MODULE SELECTION (UNICAST)
74FCT138A
U24
U25
U26
U27
+Vcc
E3
E1
E2
Y0 Y1 Y2 Y3 Y4 Y5 Y6 Y7
Y8 Y9 Y10 Y11 Y12 Y13 Y14 Y15
Y16 Y17 Y18 Y19 Y20 Y21 Y22 Y23
Y24 Y25 Y26 Y27 Y28 Y29 Y30 NC
1/8 74FCT240A
U16B
R/W
MULTICAST OPTION
74FCT138A
U28
SEL
U29
INTSEL
INTERRUPT IDENTIFICATION

PREPARED BY — ÉTABLI PAR

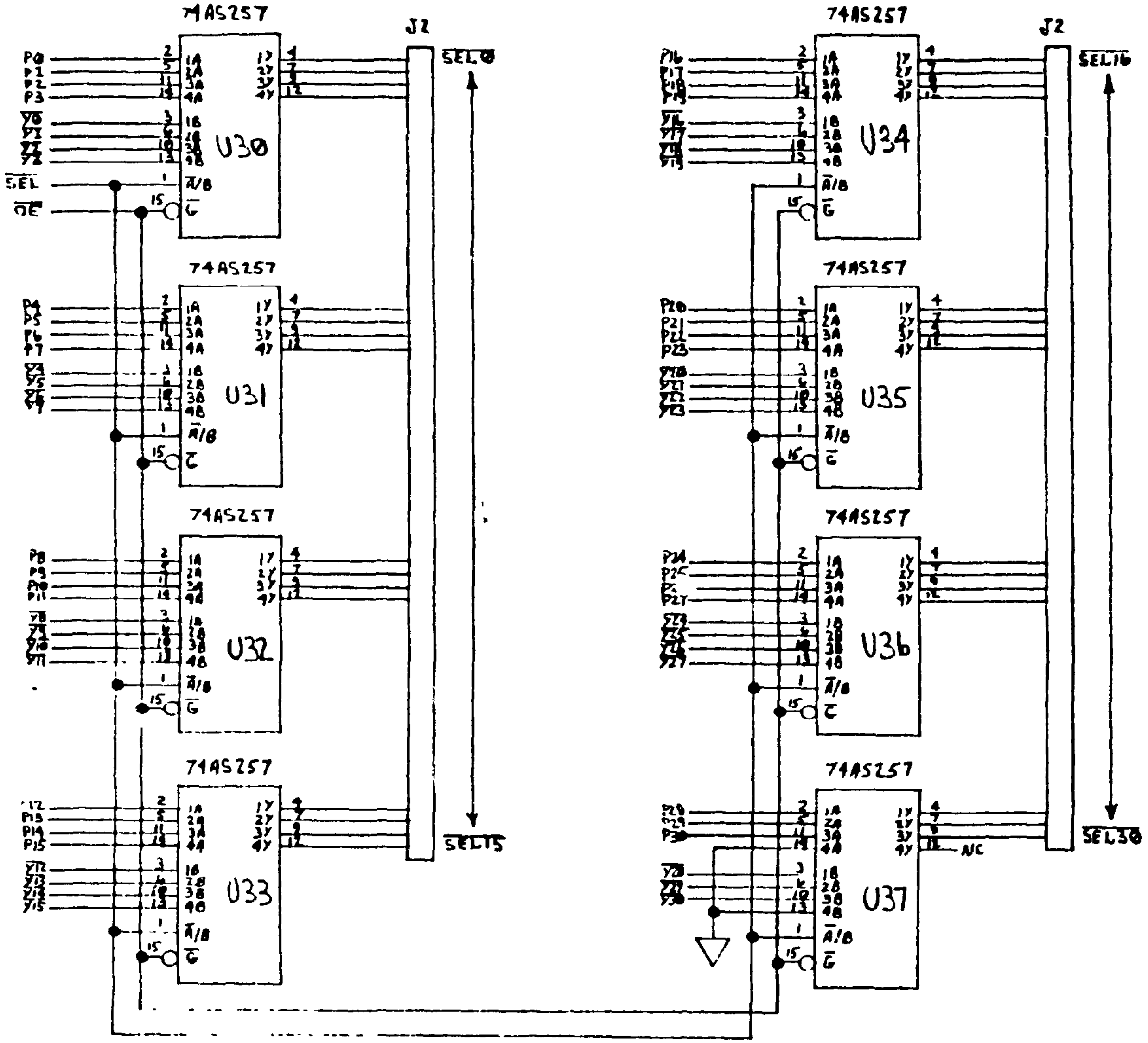
DATE

PAGE 2 OF PAGES

NAME OF SYSTEM OR PROGRAM — NOM DU SYSTÈME OU PROGRAMME

MULTICAST SYSTEM

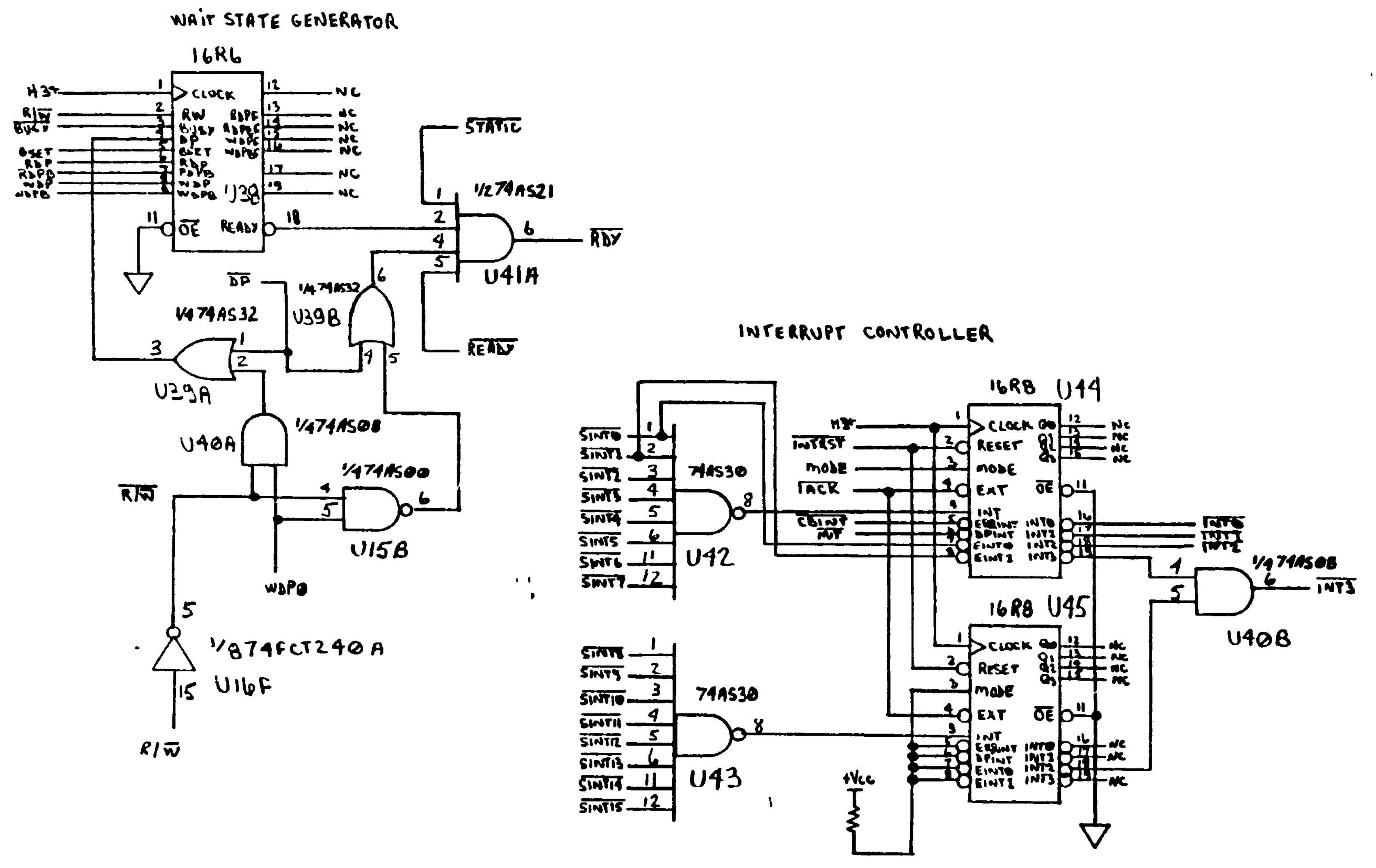
IDENTIFICATION



F

PREPARED BY – ÉTABLIE PAR	DATE	PAGE OF DE

NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME	IDENTIFICATION
WAIT STATE GENERATOR / INTERRUPT CONTROL.	



```
TITLE     Wait State Generator

;DESCRIPTION
;The wait state generator is used on the TMS320C30 DSP board.
;It generates one or two wait-states when the processor accesses
;the dual-port RAM modules. This is accomplished by sending a
.  \DY signal to the processor at the appropriate time. Four modes
;a.e available under software control. Each mode defines the type of accesses.
;A BUSY signal generated by the dual-port RAMs indicates to the DSP
;that it should extend the cycle due to an address conflict with another
;processor. The DSP when considering the BUSY must extend the access
;cycle by 30 ns to allow the slaves to be set. This extension could add
;another wait-state depending on the capacitive load on the bus and the
;driving current used from the DSP board and the dual-port RAMs module.
;
;MODE 1: read dual-port memory without considering the BUSY (1 or 2 w-s)
;MODE 2: read dual-port memory and consider the BUSY (1 or 2 wait-states)
;MODE 3: write to the d-p memory without considering the BUSY (1 or 2 w-s)
;MODE 4: write to the d-p memory and consider the BUSY (1 or 2 w-s)

PATTERN   WAIT.PDS
REVISION  1
AUTHOR    S. Martel
COMPANY   McGill University
DATE      5 August 1988

CHIP      WAIT_STATE  PAL16R6

;PINS
; 1      2     3     4     5     6     7     8     9     10
  CLOCK  RW    BUSY  DP    BSET  RDP   RDPB  WDP   WDPB  GND

; '1     12    13    14    15    16    17    18    19    20
   :     NC    RDPF  RDPBF WDPF  WDPBF NC    READY NC    VCC

;CLOCK: the PAL used H3 from the DSP the synchronize the READY signal
;RW   : conected to the R/W line (1=READ, 0=WRITE)
;BUSY : 0=access conflict 1=no conflict
;DP   : 0=access to the dual-port memory
;BSET : 0=access without considering BUSY 1=consider BUSY
;RDP  : read dual-port 0=1 wait-state 1=2 wait-states (without BUSY)
;RDPB : read dual-port 0=1 wait-state 1=2 wait-states (with BUSY)
;WDP  : write to the d-p 0=1 wait-state 1=2 wait-states (without BUSY)
;WDPB : write to the d-p 0=1 wait-state 1=2 wait-states (with BUSY)
;OE   : output enable connected to ground (hardware control)
;...F : flags with feedback to keep track of number of wait-states generated
;READY: the only output connected 0=ready 1=not ready (another wait-state)

EQUATIONS

/READY := /DP * /BSET * RW * /RDP * READY
          ;read dual-port with 1 wait-state without busy


        + /DP * /BSET * /RW * /WDP * READY
          ;write to dual-port with 1 wait-state without busy

        + /DP * BSET * RW * /RDPB * BUSY * READY
          ;read dual-port with 1 wait-state with busy

        + /DP * BSET * /RW * /WDPB * BUSY * READY
          ;write to dual-port with 1 wait-state with busy

        + /DP * /BSET * RW * RDP * /RDPF * READY
          ;read dual-port with 2 wait-states without busy
        + /DP * /BSET * /RW * WDP * /WDPF * READY
          ;write to dual-port with 2 wait-states without busy

        + /DP * BSET * RW * RDPB * /RDPBF * BUSY * READY
          ;read dual-port with 2 wait-states with busy

        + /DP * BSET * /RW * WDPB * /WDPBF * BUSY * READY
          ;write to dual-port with 2 wait-states with busy

/RDPF  := /DP * /BSET * RW * RDP
/WDPF  := /DP * /BSET * /RW * WDP
/RDPBF := /DP * BSET * RW * RDPB
/WDPBF := /DP * BSET * /RW * WDPB

SIMULATION

TRACE_ON CLOCK READY BSET RW RDP RDPB WDP WDPB DP BUSY

SETF /OE                                       ; enable outputs
     /CLOCK                                    ; initialize clock
     DP /BSET RW /RDP /RDPB /WDP /WDPB BUSY    ; initialize inputs
CLOCKF CLOCK

; 2 successive read cycles with 1 wait-state without busy tested with BUSY
SETF /DP /BUSY
CLOCKF CLOCK
CHECK /READY
CLOCKF CLOCK
CHECK READY
CLOCKF CLOCK
CHECK /READY
CLOCKF CLOCK
CHECK READY

; one read cycle with 1 wait-state with busy
SETF BSET
CLOCKF CLOCK
CHECK READY
SETF BUSY
CLOCKF CLOCK
CHECK /READY
CLOCKF CLOCK
CHECK READY

; one read cycle with 2 wait-states with busy and BUSY high
SETF RDPB
CLOCKF CLOCK
CHECK READY
CLOCKF CLOCK
CHECK /READY
CLOCKF CLOCK
CHECK READY

; one write cycle with 2 wait-states with busy and BUSY low
SETF WDPB /RW
CLOCKF CLOCK
CHECK READY
SETF /BUSY
CLOCKF CLOCK
CHECK READY
SETF BUSY
CLOCKF CLOCK
CHECK /READY
SETF RW
CLOCKF CLOCK
CHECK READY
```

```
PALASM XPLOT, V2.23 - MARKET RELEASE (2-1-88)
(C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988

Title     : Wait State Generator     Author   : S. Martel
Pattern   : WAIT.PDS                         Company : McGill University
Revision : 1                                 Date     : 5 August 1988

PAL16R6
WAIT_STATE

                11 1111 1111 2222 2222 2233
     0123 4567 8901 2345 6789 0123 4567 8901

  0 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  1 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  2 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  3 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  4 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  5 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  6 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  7 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

  8 X--- --X- -X-- -X-- -X-- ---- ---- ----
  9 -X-- --X- -X-- -X-- ---X ---- ---- ----
 10 X--- --X- -X-- -X-- ---- ---- ---X ----
 11 -X-- X-X- -X-- X--- ---- ---- ---- -X--
 12 X--- X-X- -X-- X--- ---- -X-- ---- ----
 13 -X-- X-X- -X-- X--X ---- ---- ---- ----
 14 X--- X-X- -X-- X--- ---- ---X ---- ----
 15 -X-- --X- -X-- -X-- ---- ---- -X-- ----

 16 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 17 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
    XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 19 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 20 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 21 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 22 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 23 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 24 -X-- ---- -X-- X--- ---- ---- ---- X---
 25 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 26 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 27 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 28 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 29 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 30 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 31 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 32 -X-- ---- -X-- -X-- ---- ---- X--- ----
 33 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 34 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 35 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 36 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 37 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 38 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 39 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 40 X--- ---- -X-- X--- ---- X--- ---- ----
    XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
    XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 43 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 44 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 45 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 46 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 47 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
```

```
48 X--- ---- -X-- -X-- X--- ---- ---- ----
49 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
50 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
51 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
52 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
   XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
54 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
55 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

56 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
57 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
58 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
59 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
60 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
61 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
62 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
63 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX


TOTAL FUSES BLOWN:   324
```

A

```
PALASM XPLOT, V2.23 - MARKET RELEASE (2-1-88)
(C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988

Title     : Wait State Generator          Author  : S. Martel
Pattern   : WAIT.PDS                      Company : McGill University
Revision  : 1                             Date    : 5 August 1988


PAL16R6
WAIT_STATE*
QV512*
QP20*
QF2048*
G0*F0*
L0256 01111101101110111011111111111111*
L0288 10111101101110111101111111111111*
L0320 01111101101110111111111111101111*
L0352 10110101101101111111111111111011*
L0384 01110101101101111111110111111111*
L0416 10110101101101101111111111111111*
L0448 01110101101101111111111011111111*
L0480 10111101101110111111111110111111*
L0768 10111111101101111111111111110111*
L1024 10111111101110111111111101111111*
L1280 01111111101101111101111111111111*
L1536 01111111101110110111111111111111*
V0001 C11100000N0XHHHHXHXN*
V0002 C10000000N0XHHHHXLXN*
V0003 C10000000N0XHHHHXHXN*
V0004 C10000000N0XHHHHXLXN*
V0005 C10000000N0XHHHHXHXN*
VU006 C10010000N0XHHHHXHXN*
V^007 C11010000N0XHHHHXLXN*
·   08 C11010000N0XHHHHXHXN*
VU009 C11010100N0XHLHHXHXN*
V0010 C11010100N0XHLHHXLXN*
V0011 C11010100N0XHLHHXHXN*
V0012 C01010101N0XHHHLXHXN*
V0013 C00010101N0XHHHLXHXN*
V0014 C01010101N0XHHHLXLXN*
V0015 C11010101N0XHLHHXHXN*
C2AD0*
CA03
```

```
PALASM SIMULATION, V2.23 - MARKET RELEASE (2-1-88)
(C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988
PALASM SIMULATION SELECTIVE TRACE LISTING

Title    : Wait State Generator    Author    : S. Martel
Pattern  : WAIT.PDS                Company   : McGill University
I  ision : 1                       Date      : 5 August 1988

PAL16R6
WAIT_STATE
Page :  1
      g  cg  c   c  c  cg c g  c  cg   c  c  cg   cg cg  cg   c
CLOCK LHHLLHHLHH LHHLHHLLHL LHHLHHLLHH LHHLHHLLHH LLHLLHHLLH HL
READY XXHHHHLLLH HHLLLHHHHH HHLLLHHHHH HHLLLHHHHH HHHHHHLLLL HH
BSET  LLLLLLLLLL LLLLLLLHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HH
RW    HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHLLL LLLLLLLLHH HH
RDP   LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LL
RDPB  LLLLLLLLLL LLLLLLLLLL LLLLLLLHHH HHHHHHHHHH HHHHHHHHHH HH
WDP   LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LL
WDPB  LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLHHH HHHHHHHHHH HH
DP    HHHHLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LL
BUSY  HHHHLLLLLL LLLLLLLLLL HHHHHHHHHH HHHHHHHHHH HLLLHHHHHH HH
```

```
PALASM SIMULATION, V2.23 - MARKET RELEASE (2-1-88)
(C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988
PALASM SIMULATION HISTORY LISTING

Title    : Wait State Generator    Author    : S. Martel
Pattern  : WAIT.PDS                Company   : McGill University
I  ision : 1                       Date      : 5 August 1988

PAL16R6
WAIT_STATE
Page :  1
       g  cg  c   c  c  cg c g  c  cg   c  c  cg   cg cg  cg   c
CLOCK  LHHLLHHLHH LHHLHHLLHL LHHLHHLLHH LHHLHHLLHH LLHLLHHLLH HL
RW     HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHLLL LLLLLLLLHH HH
BUSY   HHHHLLLLLL LLLLLLLLLL HHHHHHHHHH HHHHHHHHHH HLLLHHHHHH HH
DP     HHHHLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LL
BSET   LLLLLLLLLL LLLLLLLHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HH
RDP    LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LL
RDPB   LLLLLLLLLL LLLLLLLLLL LLLLLLLHHH HHHHHHHHHH HHHHHHHHHH HH
WDP    LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LL
WDPB   LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLHHH HHHHHHHHHH HH
GND    LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LL
OE     LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LL
RDPF   XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HH
RDPBF  XXHHHHHHHH HHHHHHHHHH HHHHHHHHHL LLLLLLLLLH HHHHHHHHHH LL
WDPF   XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HH
WDPBF  XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHL LLLLLLLLLL HH
READY  XXHHHHLLLH HHLLLHHHHH HHLLLHHHHH HHLLLHHHHH HHHHHHLLLL HH
VCC    HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HH
```

```
TITLE     Interrupt Controller
PATTERN   INTERRUPT.PDS
REVISION  1
AUTHOR    S. Martel
COMPANY   McGill University
DATE      23 August 1988

CHIP      INT_CNTRL  PAL16R8

;*** PINS ***

; CLK    IN     IN     IN     IN      IN    IN    IN    IN    GND   OE    OUT
; 1      2      3      4      5       6     7     8     9     10    11    12
  CLOCK RESET MODE    EXT    ERRINT DPINT EINT0 EINT1 INT   GND   OE    Q0

; OUT    OUT    OUT    OUT    OUT     OUT   OUT   VCC
; 13     14     15     16     17      18    19    20
  Q1     Q2     Q3     INT0   INT1    INT2  INT3  VCC

;*** INPUTS ***

;CLOCK  : Clock H1 running at 16.5 MHz and provided by the processor.
;RESET  : System reset (active low).
;EINT.. : External interrupts (active low).
;ERRINT : Error interrupt active low and connected to ERR from the EDAC.
;DPINT  : Dual-port memory interrupt active low.
;MODE   : Mode select. 0: Channels 0 and 1 only have interrupt facilities.
;                      1: All 8 channels have interrupt facilities.
;EXT    : One of the eight channels is read when EXT is low.
;INT    : At least one external interrupt is flag (active high).

;*** OUTPUTS ***

;  .    : 1-bit counter in order to garantee that the duration of the interrupt
;       : is at least one H1 clock cycle to be detected  by the TMS320C30.
;INT0   : Highest priority interrupt used for the error correction routine.
;INT1   : Second priority interrupt used for inter-processor communication.
;INT2   : Third priority interrupt used for channel 0 if in mode 0 or for
;         all 8 channels if in mode 1.
;INT3   : Lowest priority interrupt used for channel 1 if in mode0.

EQUATIONS

/INT0 := Q0 * /ERRINT * RESET
/Q0   := /ERRINT * RESET

/INT1 := Q1 * /DPINT * RESET
/Q1   := /DPINT * RESET

/INT2 := Q2 * /MODE * /EINT0 * RESET
       + Q2 * MODE * INT * EXT * RESET
/Q2   := /MODE * /EINT0 * RESET
       + MODE * INT * EXT * RESET

/INT3 := Q3 * /MODE * /EINT1 * RESET
/Q3   := /MODE * /EINT1 * RESET

SIMULATION

TRACE_ON RESET CLOCK MODE EXT ERRINT DPINT EINT0 EINT1 INT INT0 INT1 INT2
         INT3

SETF /OE                                      ;Enable outputs
     /CLOCK                                   ;Initialize clock
     /RESET EXT ERRINT DPINT EINT0 EINT1 /INT ;Initialize inputs
     /MODE
```

```
CLOCKF CLOCK

; Error interrupt during reset
SETF /ERRINT
CLOCKF CLOCK
CHECK INT0

; 2 consecutive error interrupts
SETF RESET
CLOCKF CLOCK
CHECK /INT0
CLOCKF CLOCK
CHECK INT0
CLOCKF CLOCK
CHECK INT0
SETF ERRINT
CLOCKF CLOCK
CHECK INT0
SETF /ERRINT
CLOCKF CLOCK
CHECK /INT0
CLOCKF CLOCK
CHECK INT0
SETF ERRINT
CLOCKF CLOCK
CHECK INT0

; Error interrupt with dual-port memory interrupt
SETF /ERRINT /DPINT
CLOCKF CLOCK
CHECK /INT0 /INT1 INT2 INT3
CLOCKF CLOCK
CHECK INT0 INT1 INT2 INT3
CLOCKF CLOCK
CHECK INT0 INT1 INT2 INT3
CLOCKF CLOCK
CHECK INT0 INT1 INT2 INT3

; External interrupt from channel 0 in mode 0
SETF /EINT0
CLOCKF CLOCK
CHECK /INT2 INT3
CLOCKF CLOCK
CHECK INT2
SETF EINT0
CLOCKF CLOCK
CHECK INT2

; External interrupt from channel 1 and 3 in mode 1
SETF /EINT1 INT MODE
CLOCKF CLOCK
CHECK /INT2 INT3
CLOCKF CLOCK
CHECK INT2 INT3
CLOCKF CLOCK
CHECK INT2 INT3
SETF /EXT
CLOCKF CLOCK
CHECK INT2 INT3
SETF EXT
CLOCKF CLOCK
CHECK /INT2 INT3
CLOCKF CLOCK
CHECK INT2 INT3
```

```
PALASM XPLOT, V2.23 - MARKET RELEASE (2-1-88)
(C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988

Title     : Interrupt Controller      Author  : S. Martel
Pattern   : INTERRUPT.PDS                     Company : McGill University
Revision  : 1                                 Date    : 23 August 1988

PAL16R8
INT_CNTRL

                 11 1111 1111 2222 2222 2233
   0123 4567 8901 2345 6789 0123 4567 8901

 0 X--- -X-- ---- ---- --X- ---- -X-- ----
 1 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 2 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 3 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 4 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 5 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 6 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 7 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 8 X--- X--- X--- ---- ---- --X- ---- X---
 9 X--- -X-- ---- ---- ---- -XX- ---- ----
10 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
11 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
12 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
13 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
14 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
15 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

16 X--- ---- ---- ---- -X-- ---- --X- ----
17 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
   XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
19 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
20 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
21 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
22 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
23 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

24 X--- ---- ---- -X-- ---- ---- ---- --X-
25 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
26 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
27 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
28 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
29 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
30 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
31 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

32 X--- -X-- ---- ---- ---- ---- -X-- ----
33 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
34 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
35 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
36 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
37 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
38 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
39 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

40 X--- X--- X--- ---- ---- ---- ---- X---
   X--- -X-- ---- ---- ---- -X-- ---- ----
42 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
43 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
44 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
45 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
46 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
47 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
```

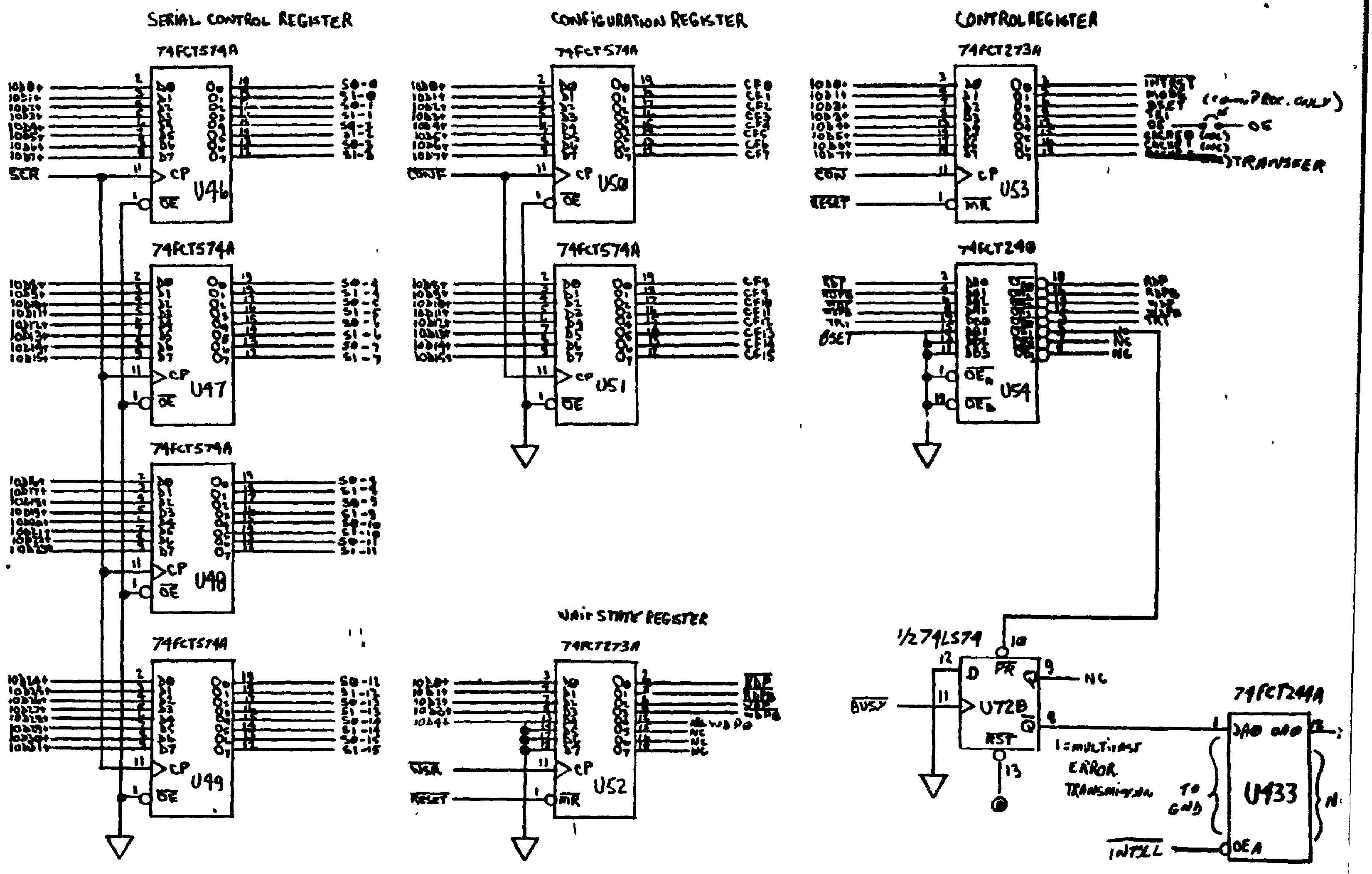
```
48 X--- ---- ---- ---- -X-- ---- ---- ----
49 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
50 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
51 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
52 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
   XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
54 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
55 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

56 X--- ---- ---- -X-- ---- ---- ---- ----
57 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
58 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
59 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
60 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
61 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
62 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
63 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX


TOTAL FUSES BLOWN:   287
```

```
PALASM XPLOT, V2.23 - MARKET RELEASE (2-1-88)
(C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988

Title     : Interrupt Controller          Author  : S. Martel
Pattern   : INTERRUPT.PDS                 Company : McGill University
Revision  : 1                             Date    : 23 August 1988


PAL16R8
INT CNTRL*
QV5I2*
QP20*
QF2048*
G0*F0*
L0000 01111011111111111110111111011111*
L0256 01110111011111111111011111110111*
L0288 01111011111111111111001111111111*
L0512 01111111111111111110111111101111*
L0768 01111111111110111111111111111101*
L1024 01111011111111111111111110111111*
L1280 01110111011111111111111111110111*
L1312 01111011111111111111110111111111*
L1536 01111111111111111110111111111111*
L1792 01111111111110111111111111111111*
V0001 C0011111ONOHHHHHHHHN*
V0002 C0010111ONOHHHHHHHHN*
V0003 C1010111ONOLHHHLHHHN*
V0004 C1010111ONOLHHHHHHHN*
V0005 C1010111ONOLHHHHHHHN*
V0006 C1011111ONOHHHHHHHHN*
V0007 C1010111ONOLHHHLHHHN*
V0008 C1010111ONOLHHHHHHHN*
V0009 C1011111ONOHHHHHHHHN*
V0010 C1010011ONOLLHHLLHHN*
V0011 C1010011ONOLLHHHHHHN*
V0012 C1010011ONOLLHHHHHHN*
V0013 C1010011ONOLLHHHHHHN*
V0014 C1010001ONOLLLHHHLHN*
V0015 C1010001ONOLLLHHHHHN*
V0016 C1010011ONOLLHHHHHHN*
V0017 C1110010lNOLLLHHHLHN*
V0018 C1110010lNOLLLHHHHHN*
V0019 C1110010lNOLLLHHHHHN*
V0020 C1100010lNOLLHHHHHHN*
V0021 C1110010lNOLLLHHHLHN*
V0022 C1110010lNOLLLHHHHHN*
C25BC*
E4A3
```

```
PALASM SIMULATION, V2.23 - MARKET RELEASE (2-1-88)
(C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988
PALASM SIMULATION SELECTIVE TRACE LISTING

Title     : Interrupt Controller     Author    : S. Martel
Pattern   : INTERRUPT.PDS            Company   : McGill University
Revision  : 1                        Date      : 23 August 1988

PAL16R8
INT_CNTRL
Page :  1
        g  cg cg   c  c cg  c g  c  cg   cg  c  c c
 RESET  LLLLLLLHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 CLOCK  LHHLLHLLHH LHHLHLLHHL LHHLHHLLHH LLHHLHHLHL
 MODE   LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 EXT    HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 ERRINT HHHHLLLLLL LLLLLLHHHH LLLLLLLHHH HLLLLLLLLL
 DPINT  HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HLLLLLLLLL
 EINT0  HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 EINT1  HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 INT    LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 INT0   XXHHHHHHHL LLHHHHHHHH HHLLLHHHHH HHHLLLHHHH
 INT1   XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHLLLHHHH
 INT2   XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 INT3   XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH


PAL16R8
INT_CNTRL
Page :  2
         cg  c  cg   cg  c  c  cg  cg  c   c
 RESET  HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHH
 CLOCK  HLLHHLHHLL HHLLHHLHHL HLLHHLLHHL HHL
 MODE   LLLLLLLLLL LLLHHHHHHH HHHHHHHHHH HHH
 EXT    HHHHHHHHHH HHHHHHHHHH HHLLLLHHHH HHH
 ERRINT LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLL
 DPINT  LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLL
 EINT0  HHLLLLLLLH HHHHHHHHHH HHHHHHHHHH HHH
 EINT1  HHHHHHHHHH HHHLLLLLLL LLLLLLLLLL LLL
 INT    LLLLLLLLLL LLLHHHHHHH HHHHHHHHHH HHH
 INT0   HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHH
 INT1   HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHH
 INT2   HHHHLLLHHH HHHHLLLLHH HHHHHHHHLL LHH
 INT3   HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHH
```

```
PALASM SIMULATION, V2.23 - MARKET RELEASE (2-1-88)
(C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988
PALASM SIMULATION HISTORY LISTING

Title     : Interrupt Controller     Author    : S. Martel
Pattern   : INTERRUPT.PDS            Company   : McGill University
   ision  : 1                        Date      : 23 August 1988

PAL16R8
INT_CNTRL
Page :  1
          g  cg cg   c  c cg  c g  c  cg   cg   c c c
 CLOCK    LHHLLHLLHH LHHLHLLHHL LHHLHHLLHH LLHHLHHLHL
 RESET    LLLLLLLHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 MODE     LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 EXT      HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 ERRINT   HHHHLLLLLL LLLLLLHHHH LLLLLLLHHH HLLLLLLLLL
 DPINT    HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HLLLLLLLLL
 EINT0    HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 EINT1    HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 INT      LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 GND      LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 OE       LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 Q0       XXHHHHHHHL LLLLLLLLHH HHLLLLLLLH HHHLLLLLLL
 Q1       XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHLLLLLLL
 Q2       XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 Q3       XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 INT0     XXHHHHHHHL LLHHHHHHHH HHLLLHHHHH HHHLLLHHHH
 INT1     XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHLLLHHHH
 INT2     XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 INT3     XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 VCC      HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH


PAL16R8
INT_CNTRL
Page :  2
           cg   c cg   cg  c  c  cg  cg  c   c
 CLOCK    HLLHHLHHLL HHLLHHLHHL HLLHHLLHHL HHL
 RESET    HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHH
   DE     LLLLLLLLLL LLLHHHHHHH HHHHHHHHHH HHH
  XT      HHHHHHHHHH HHHHHHHHHH HHLLLLHHHH HHH
 ERRINT   LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLL
 DPINT    LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLL
 EINT0    HHLLLLLLLH HHHHHHHHHH HHHHHHHHHH HHH
 EINT1    HHHHHHHHHH HHHLLLLLLL LLLLLLLLLL LLL
 INT      LLLLLLLLLL LLLHHHHHHH HHHHHHHHHH HHH
 GND      LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLL
 OE       LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLL
 Q0       LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLL
 Q1       LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLL
 Q2       HHHHLLLLLL LHHHHLLLLL LLLLHHHHLL LLL
 Q3       HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHH
 INT0     HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHH
 INT1     HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHH
 INT2     HHHHLLLHHH HHHHHLLLHH HHHHHHHHLL LHH
 INT3     HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHH
 VCC      HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHH
```

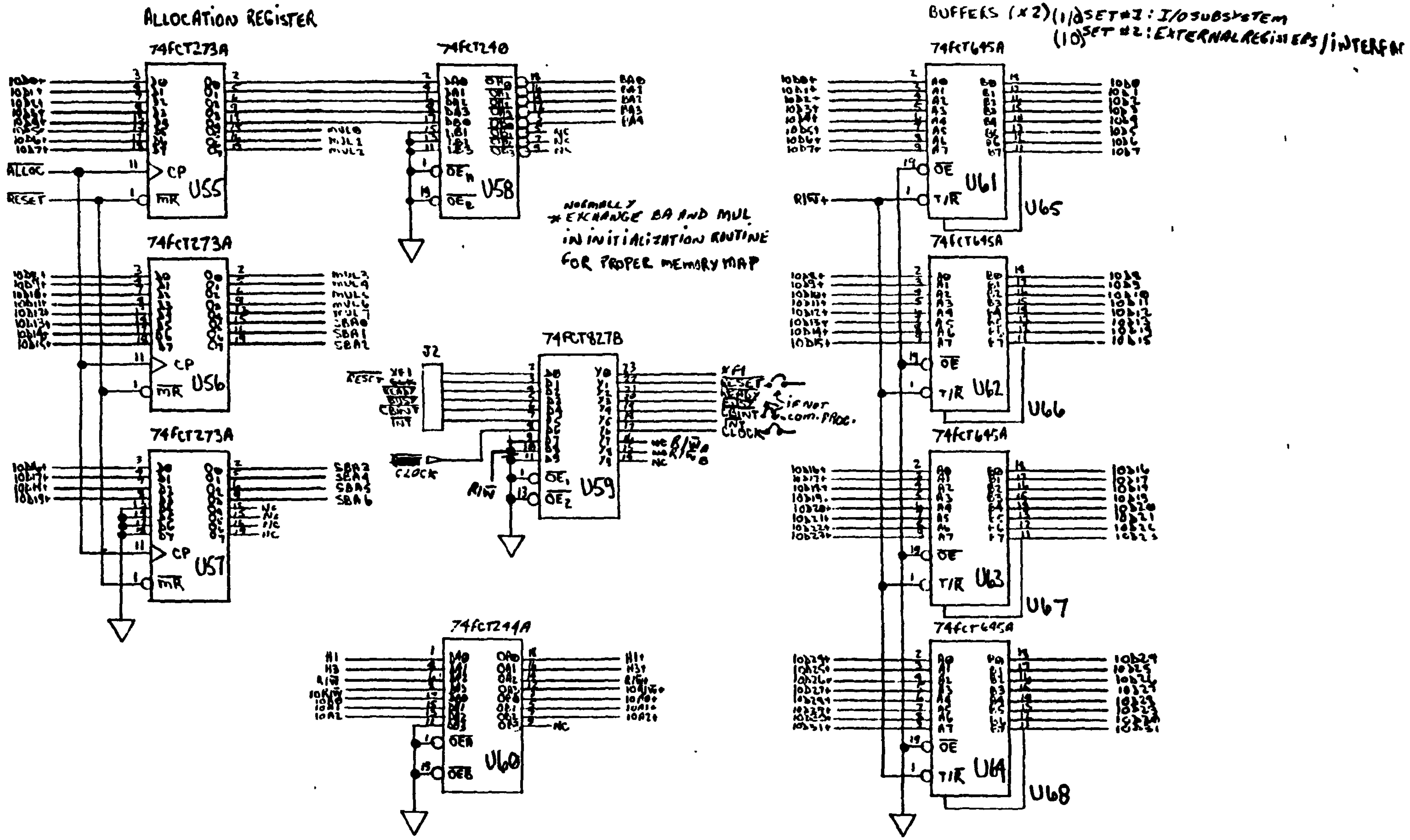
PREPARED BY – ÉTABLI PAR	DATE	PAGE	PAGE
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME EXTERNAL REGISTERS	IDENTIFICATION		



PREPARED BY – ÉTABLIE PAR | DATE | PAGE OF DE | PAGE

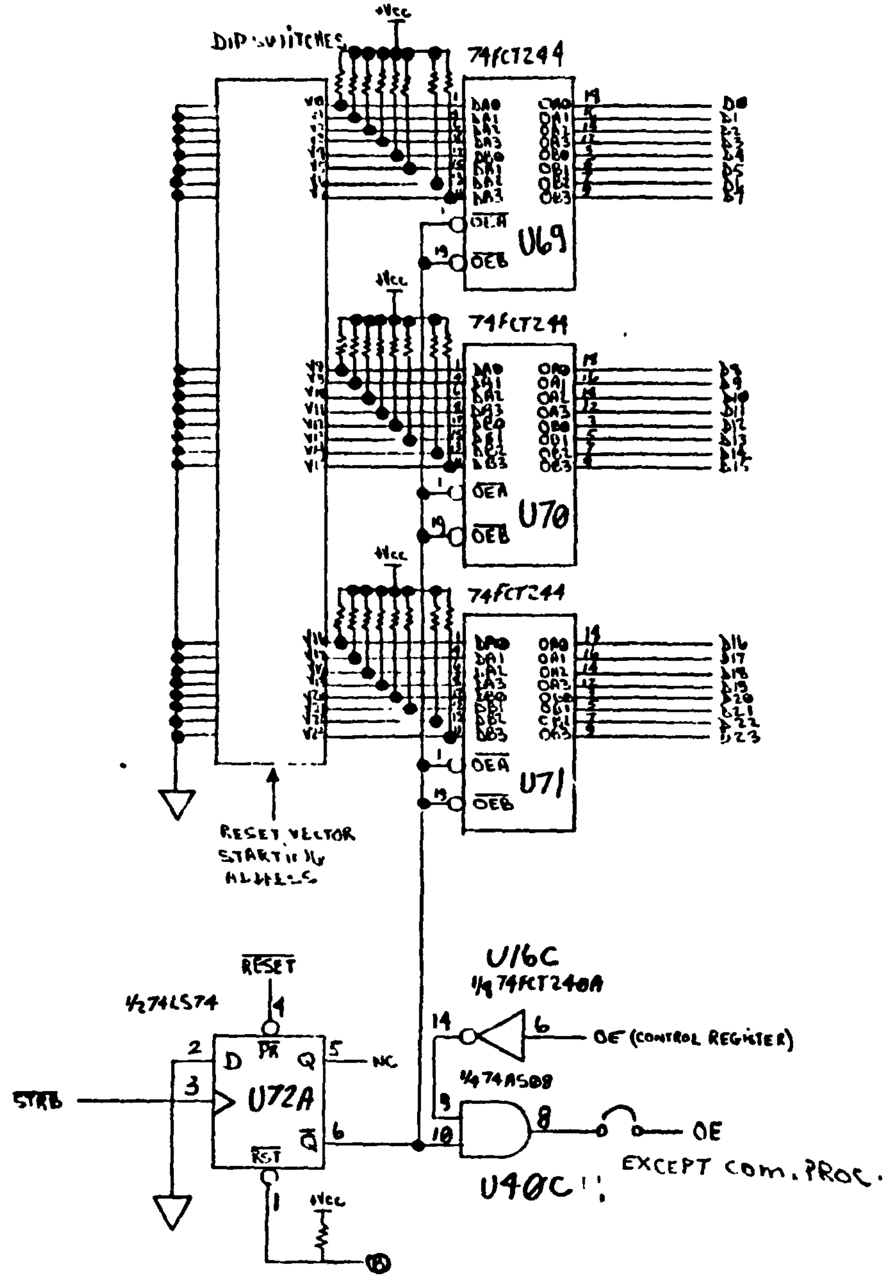
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME | IDENTIFICATION

MISCELLANEOUS



NORMALLY
* EXCHANGE BA AND MUL IN INITIALIZATION ROUTINE FOR PROPER MEMORY MAP

PREPARED BY – ÉTABLI PAR	DATE	PAGE OF DE	PAGE
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME START-UP CIRCUITRY		IDENTIFICATION	



APPENDIX B

LOCAL MEMORY BOARD

Sheet B1 - Static RAM interface (first half memory array)
Sheet B2 - Static RAM interface (second half memory array)
Sheet B3 - Static RAM block (memory banks 0 to 3)
Sheet B4 - Static RAM block (memory banks 4 to 7)

Local memory

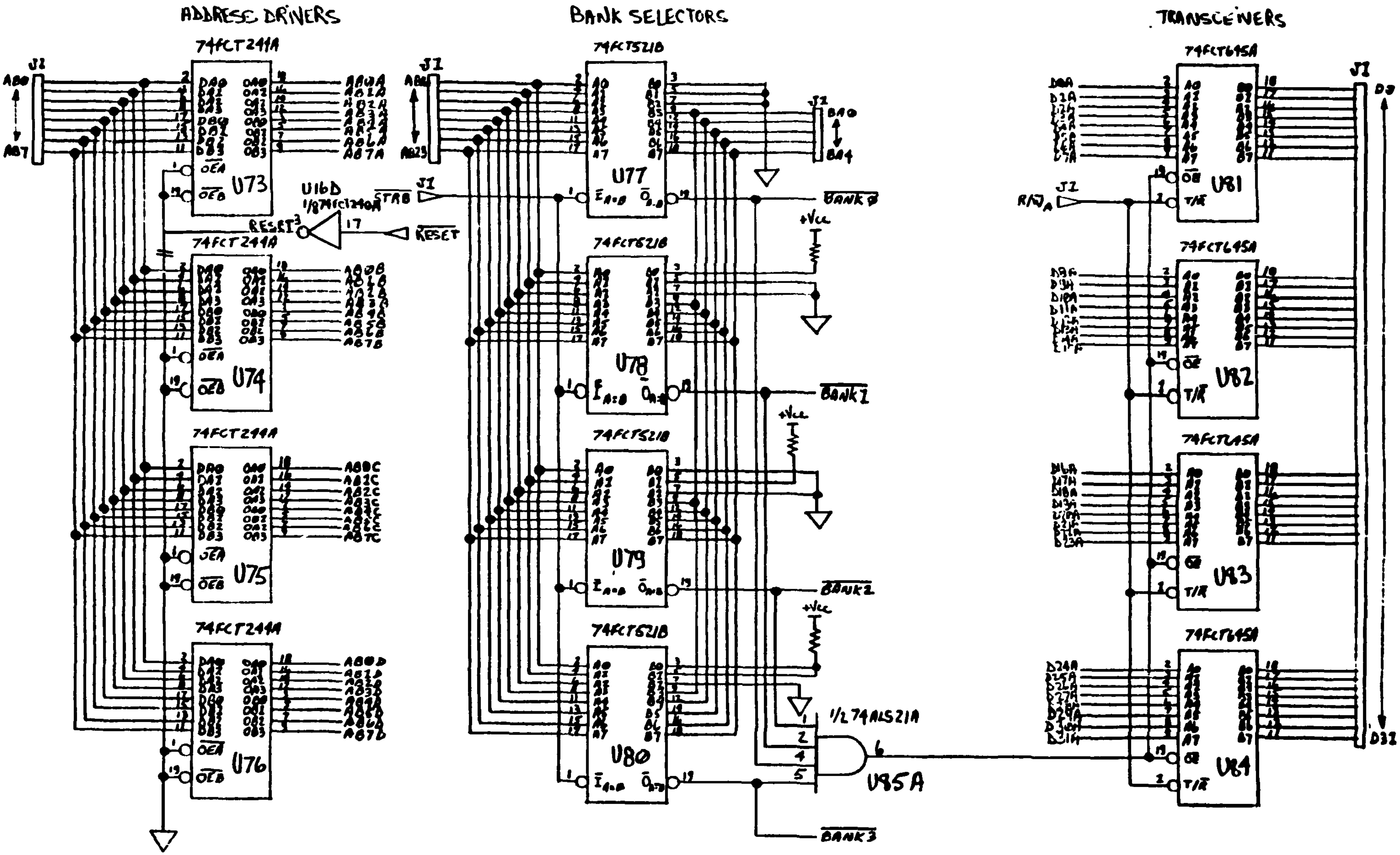
Type	Description	Quantity
74FCT244A	Fast CMOS octal buffer/line driver	8
74FCT521B	Fast CMOS 8-bit identity comparator	8
74FCT645A	Fast CMOS non-inverting buffer transceiver	8
74ALS21A	Dual 4-input positive-AND gates	1
MT5C2564	65,536 x 4 bit static RAM (25 ns access)	64
Total	89	

PREPARED BY – ETABLI PAR | DATE | PAGE 1 OF | PAGES

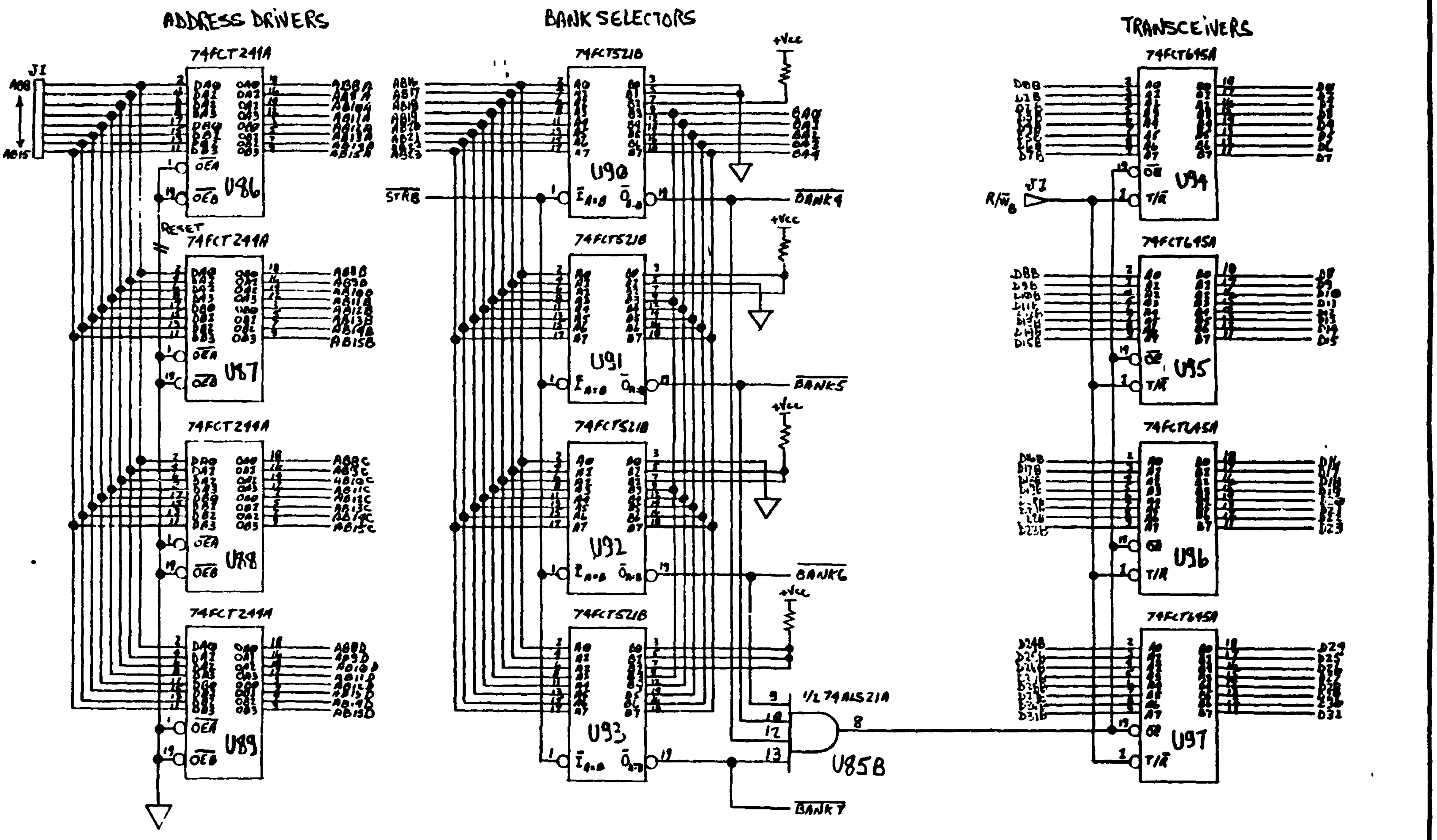
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTEME OU PROGRAMME

SRAM INTERFACE

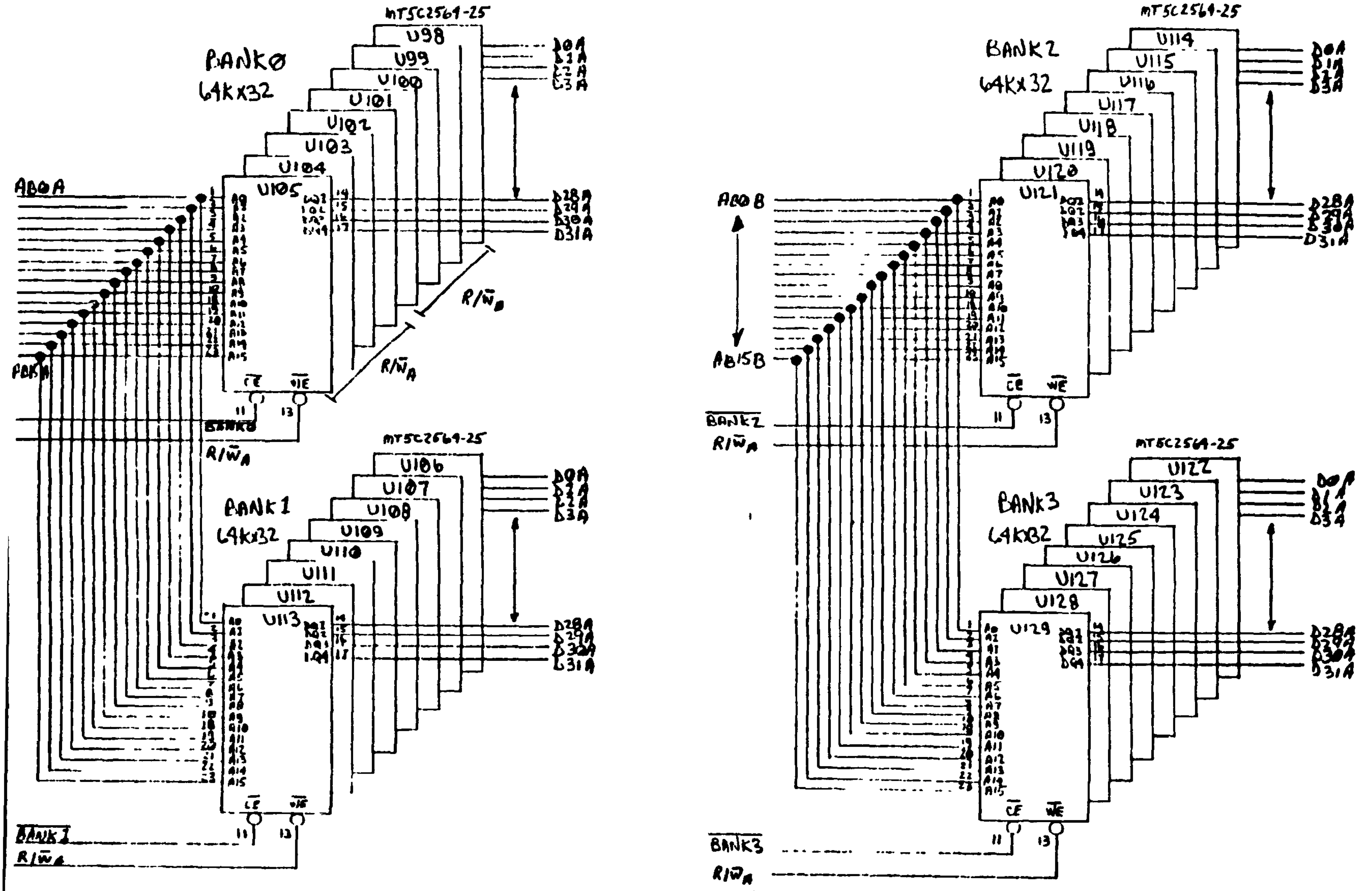
IDENTIFICATION



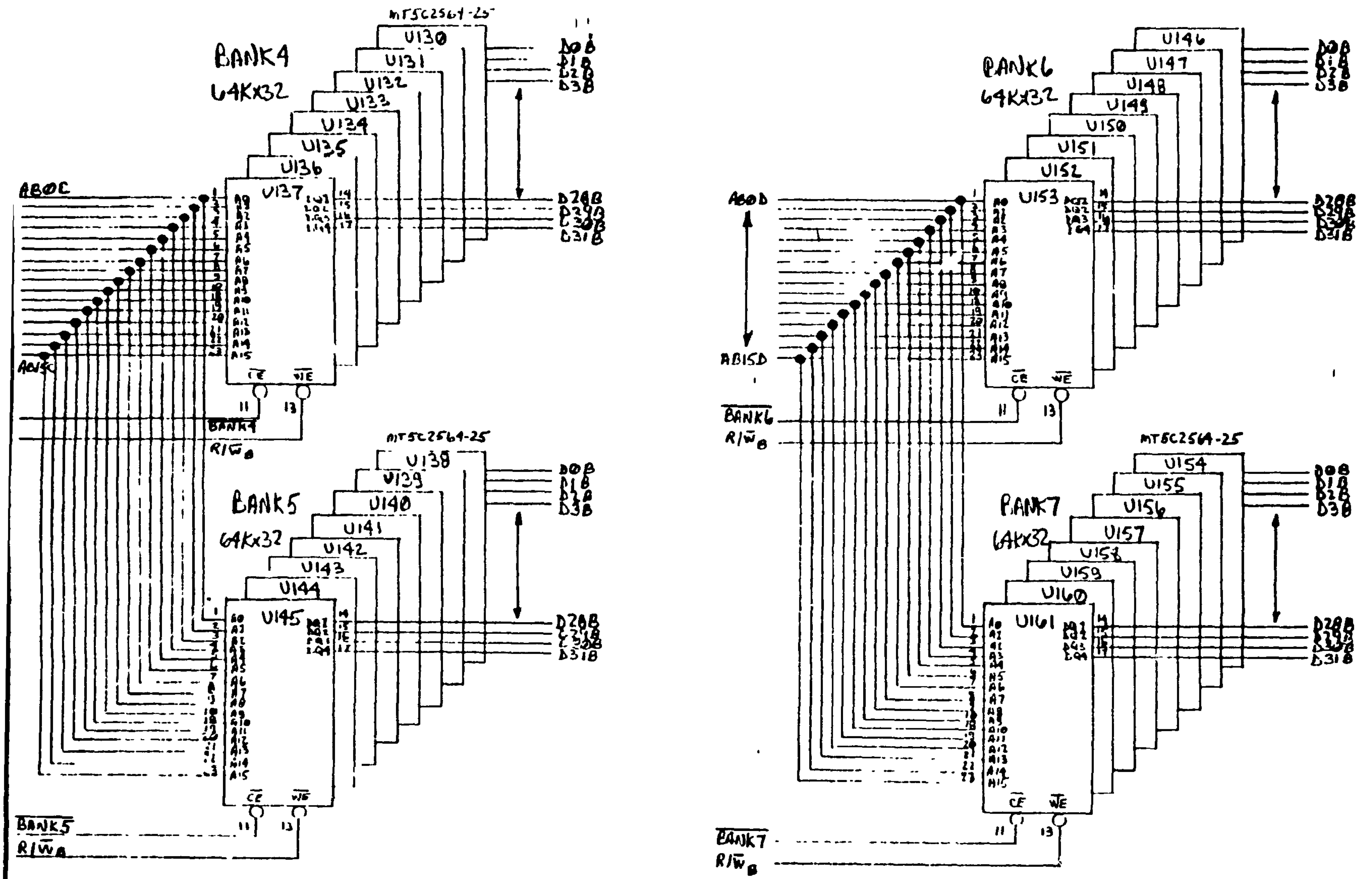
PREPARED BY – ÉTABLI PAR	DATE	PAGE 2 OF DE	PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME SRAM INTERFACE	IDENTIFICATION		



PREPARED BY – ÉTABLI PAR	DATE	PAGE 1 OF	PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME SRAM BLOCK	IDENTIFICATION		



PREPARED BY – ÉTABLI PAR	DATE	PAGE 2 OF DE	PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME		IDENTIFICATION	



E4

APPENDIX C

DUAL-PORT MODULE

Sheet C1 - Master dual-port RAM and data byte 0 transceivers

Sheet C2 - Slave dual-port RAM and data byte 1 transceivers

Sheet C3 - Slave dual-port RAM and data byte 2 transceivers

Sheet C4 - Slave dual-port RAM and data byte 3 transceivers

Sheet C5 - Dual-port RAM module interface

Dual-port RAM module

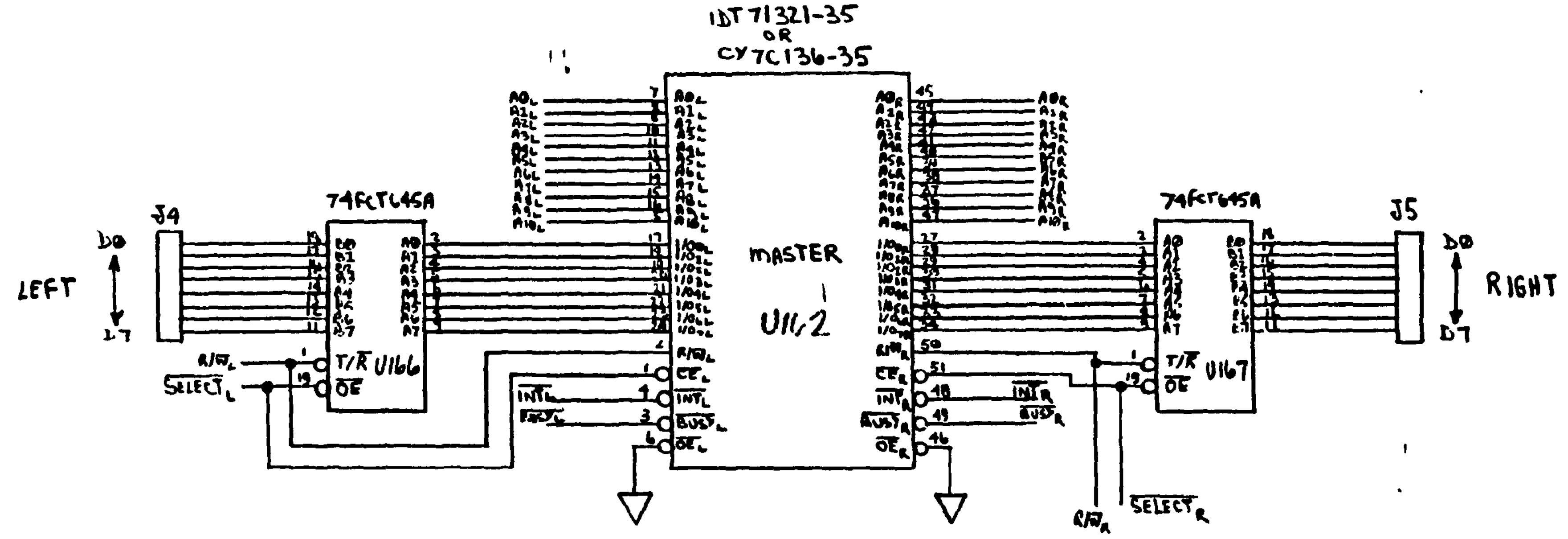
Type	Description	Quantity
IDT71321	CMOS dual-port RAM (2K x 8-bit) (master)	1
IDT71421	CMOS dual-port RAM (2K x 8-bit) (slave)	3
74FCT645A	Fast CMOS non-inverting buffer transceiver	8
74FCT244A	Fast CMOS octal buffer/line driver	3
74FCT240A	Fast CMOS octal buffer/line driver	1
74ALS09	Quad. 2-input pos.AND gates with open-collector	1

Total 17

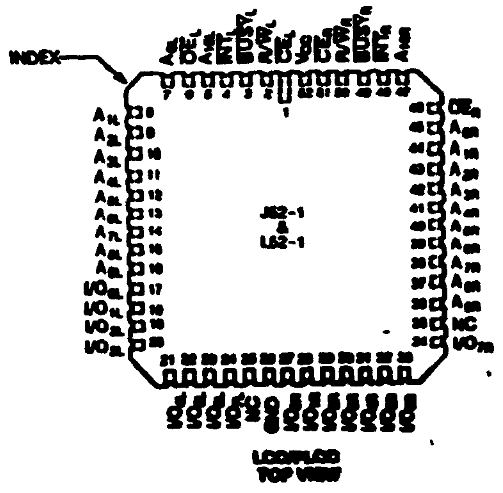
PREPARED BY – ÉTABLI PAR | DATE | PAGE 1 OF

NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME | IDENTIFICATION

DUAL-PORT MODULE

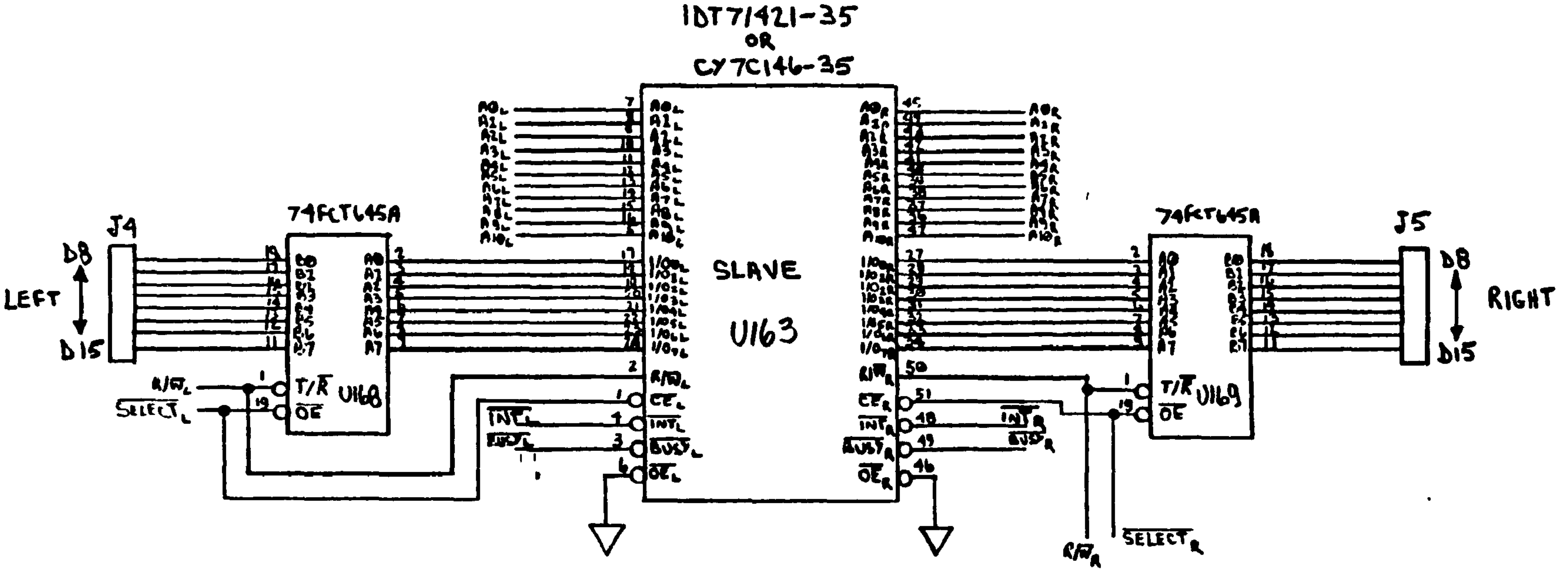


PIN CONFIGURATIONS

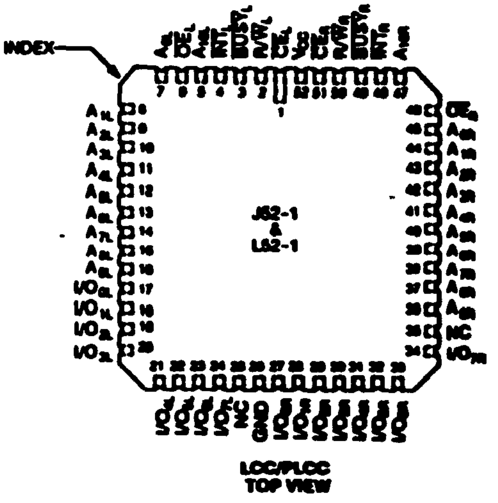


C1

PREPARED BY – ÉTABLIE PAR	DATE	PAGE 2 OF
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME		IDENTIFICATION
DUAL-PORT MODULE		

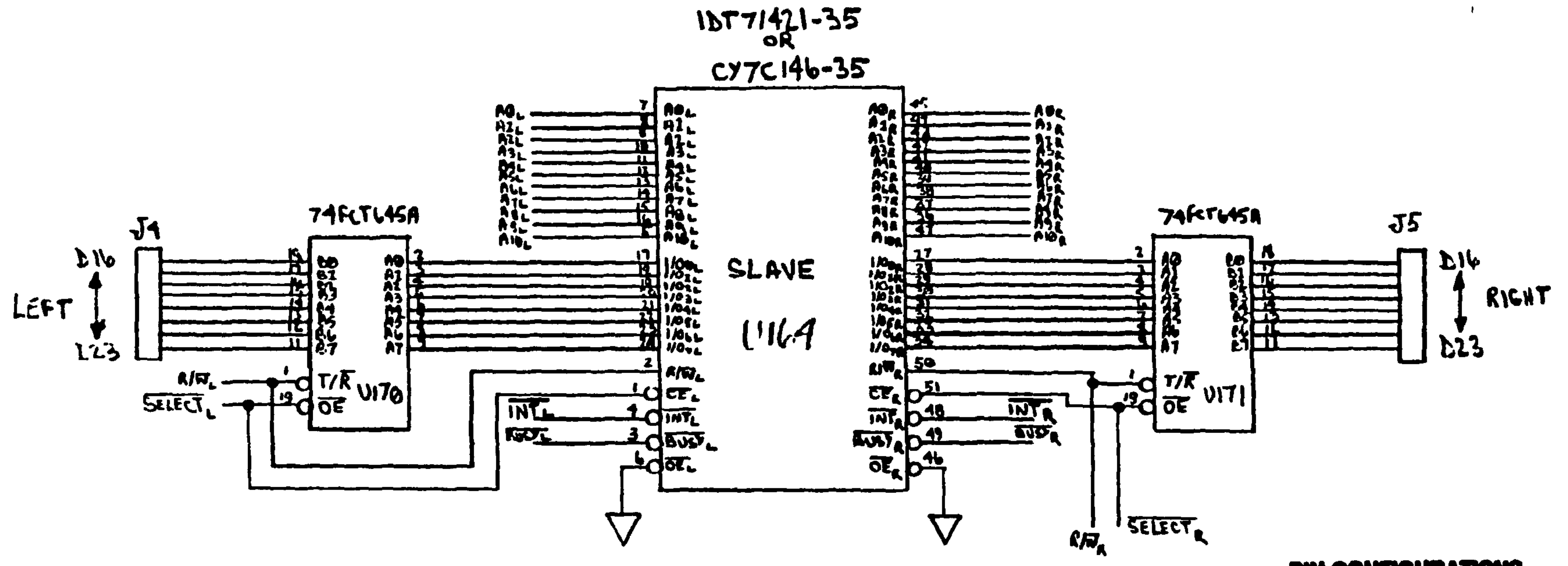

IDT71421-35
OR
CY7C146-35
SLAVE
U163
74FCT645A
U168
74FCT645A
U169
J4
J5
D8
D15
LEFT
RIGHT
T/R̄
OE
R/W̄L
SELECTL
R/W̄R
SELECTR
INTL
BUSYL
INTR
BUSYR
CEL
OEL
CER
OER
A0L A1L A2L A3L A4L A5L A6L A7L A8L A9L A10L
A0R A1R A2R A3R A4R A5R A6R A7R A8R A9R A10R
I/O0L I/O1L I/O2L I/O3L I/O4L I/O5L I/O6L I/O7L
I/O0R I/O1R I/O2R I/O3R I/O4R I/O5R I/O6R I/O7R
A0 A1 A2 A3 A4 A5 A6 A7
B0 B1 B2 B3 B4 B5 B6 B7

PIN CONFIGURATIONS

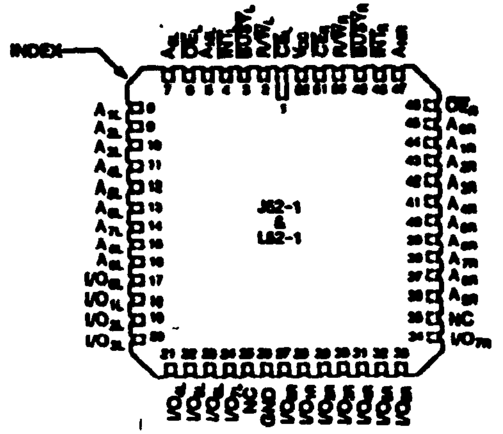


C2

PREPARED BY – ÉTABLI PAR	DATE	PAGE 3 OF	PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME DUAL-PORT MODULE		IDENTIFICATION	



PIN CONFIGURATIONS



LCC/PLCC
TOP VIEW

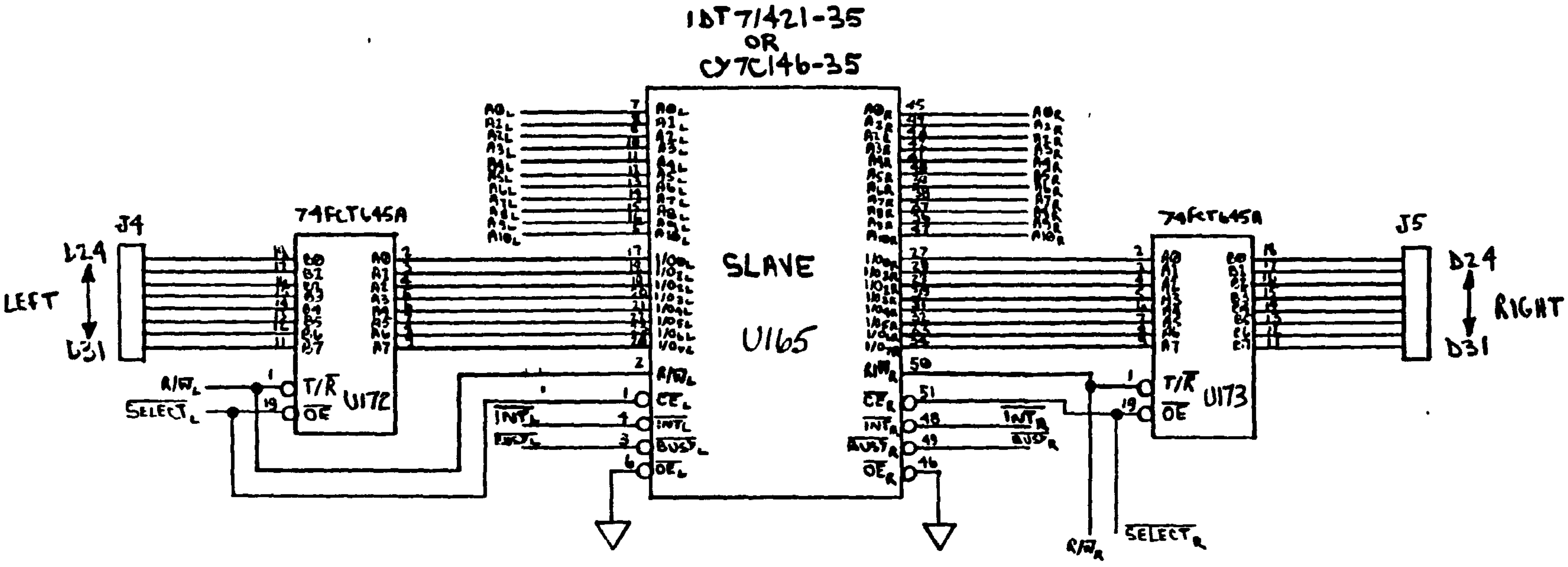
PREPARED BY – ÉTABLI PAR

PAGE 4 OF PAGES

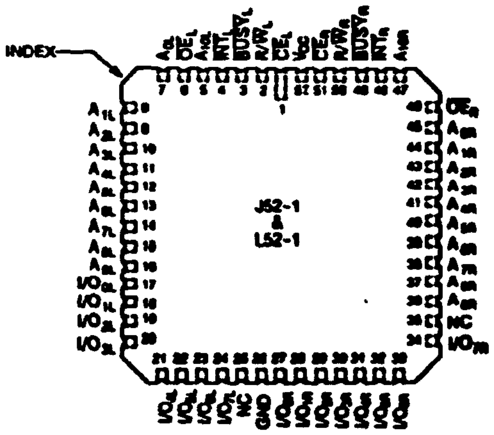
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME

DUAL-PORT MODULE

IDENTIFICATION



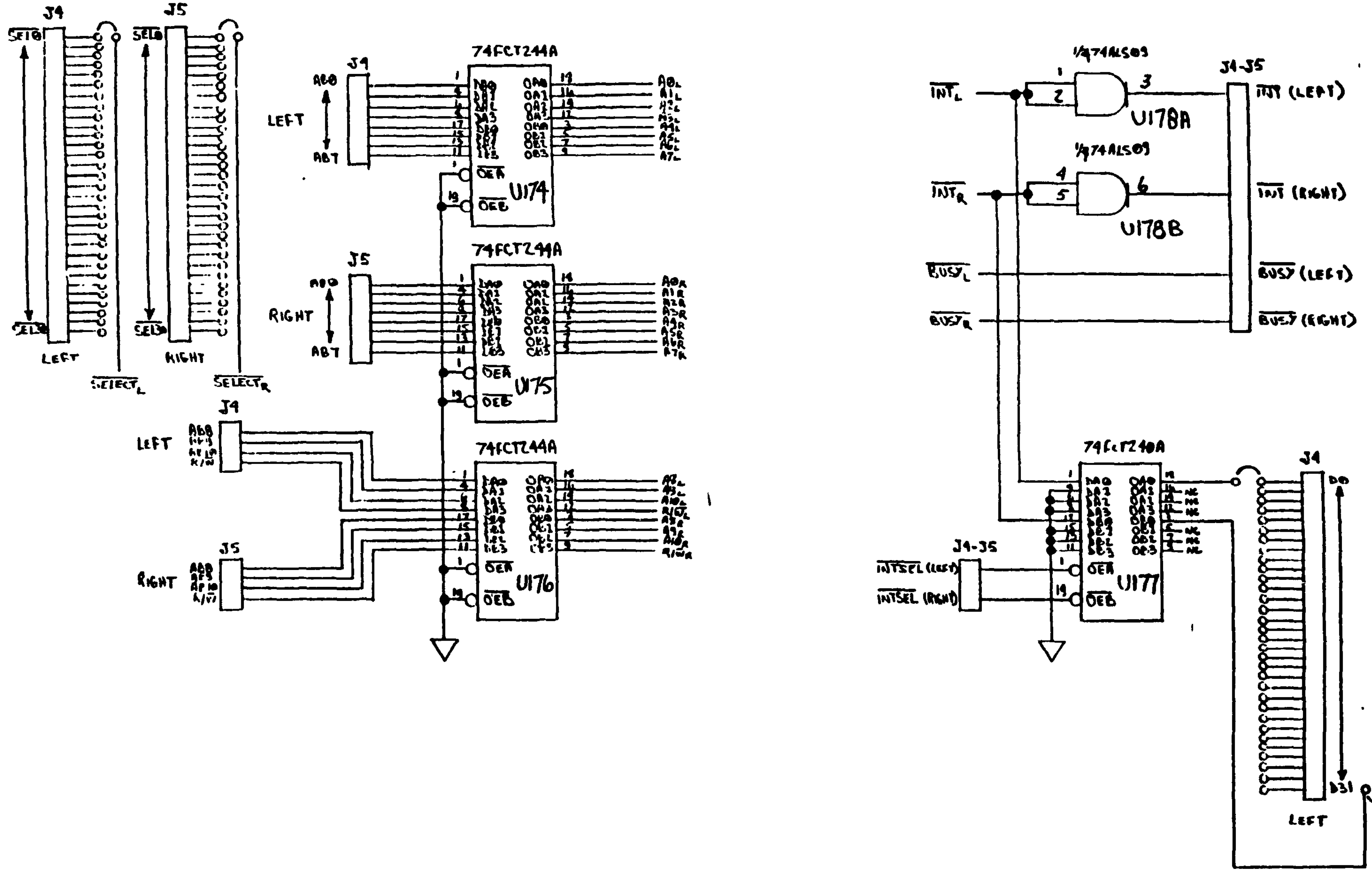
PIN CONFIGURATIONS



LCC/PLCC
TOP VIEW

PREPARED BY – ÉTABLI PAR	DATE	PAGE	PAGES
		5 OF	

NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME	IDENTIFICATION
DUAL-PORT MODULE	



APPENDIX D

I/O SUBSYSTEM

Sheet D1 - Serial channels 0 and 1
Sheet D2 - Serial channels 2 and 3
Sheet D3 - Serial channels 4 and 5
Sheet D4 - Serial channels 6 and 7
Sheet D5 - Serial channels 8 and 9
Sheet D6 - Serial channels 10 and 11
Sheet D7 - Serial channels 12 and 13
Sheet D8 - Serial channels 14 and 15
Sheet D9 - Serial channel decode
Sheet D10 - Serial control block. control lines 0 to 7
Sheet D11 - Serial control block. control lines 8 to 15
Sheet D12 - Serial control block. control lines 16 to 23
Sheet D13 - Serial control block. control lines 24 to 31
Sheet D14 - Serial control block registers (polarity and pulse state registers)
Sheet D15 - Serial control block / start control pulses circuitry
Sheet D16 - RS422 interface. drivers for data out and clock out
Sheet D17 - RS422 interface. drivers for latch enable and controls 0 to 7
Sheet D18 - RS422 interface. drivers for controls 8 to 15
Sheet D19 - RS422 interface. receivers for data in and clock in
Sheet D20 - interrupt circuitry (mask, interrupt, and status registers)

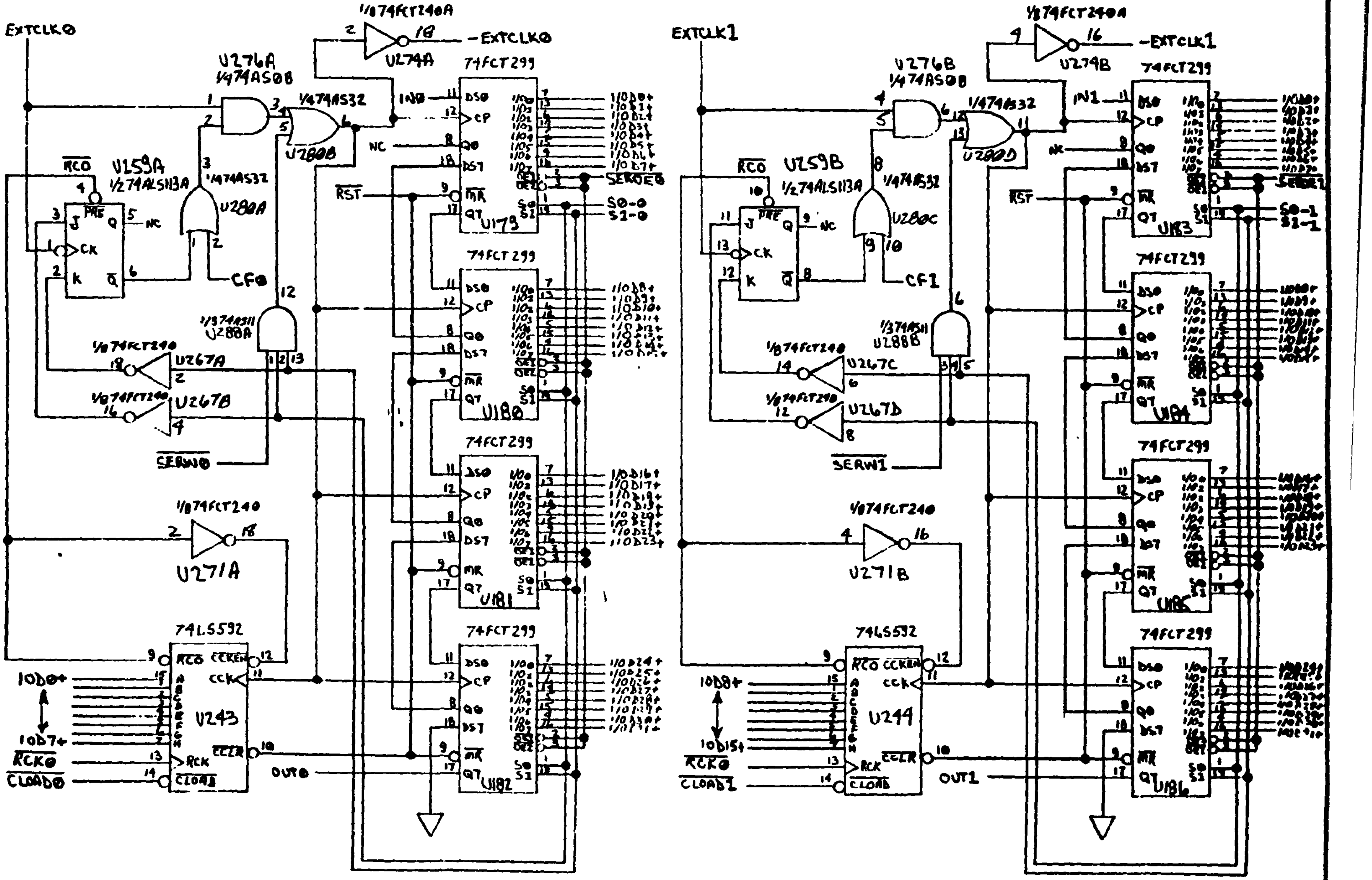
I/O subsystem

Type	Description	Quantity
74FCT299	Fast CMOS universal shift register	64
74LS592	8-bit binary counter with input register	48
74ALS113A	Dual J-K negative-edge-triggered flip-flops	8
74FCT240	Fast CMOS octal buffer/line driver	11
74FCT240A	Fast CMOS octal buffer/line driver	2
74F08	Quadruple 2-input positive-AND gates	4
74F32	Quadruple 2-input positive-OR gates	8
74F11	Triple 3-input positive-AND gates	6
74FCT521B	Fast CMOS 8-bit identity comparator	1
74FCT138A	Fast CMOS 1-of-8 decoder	7
74ALS86	Quadruple 2-input exclusive-OR gates	8
74FCT574A	Fast CMOS octal D register (3-state)	14
74ALS32	Quadruple 2-input positive-OR gates	12
AM26LS31C	Quadruple differential line drivers	20
AM26LS33AC	Quadruple differential line receivers	8
Total	221	

PREPARED BY – ÉTABLIE PAR | DATE | PAGE 1 OF DE | PAGES

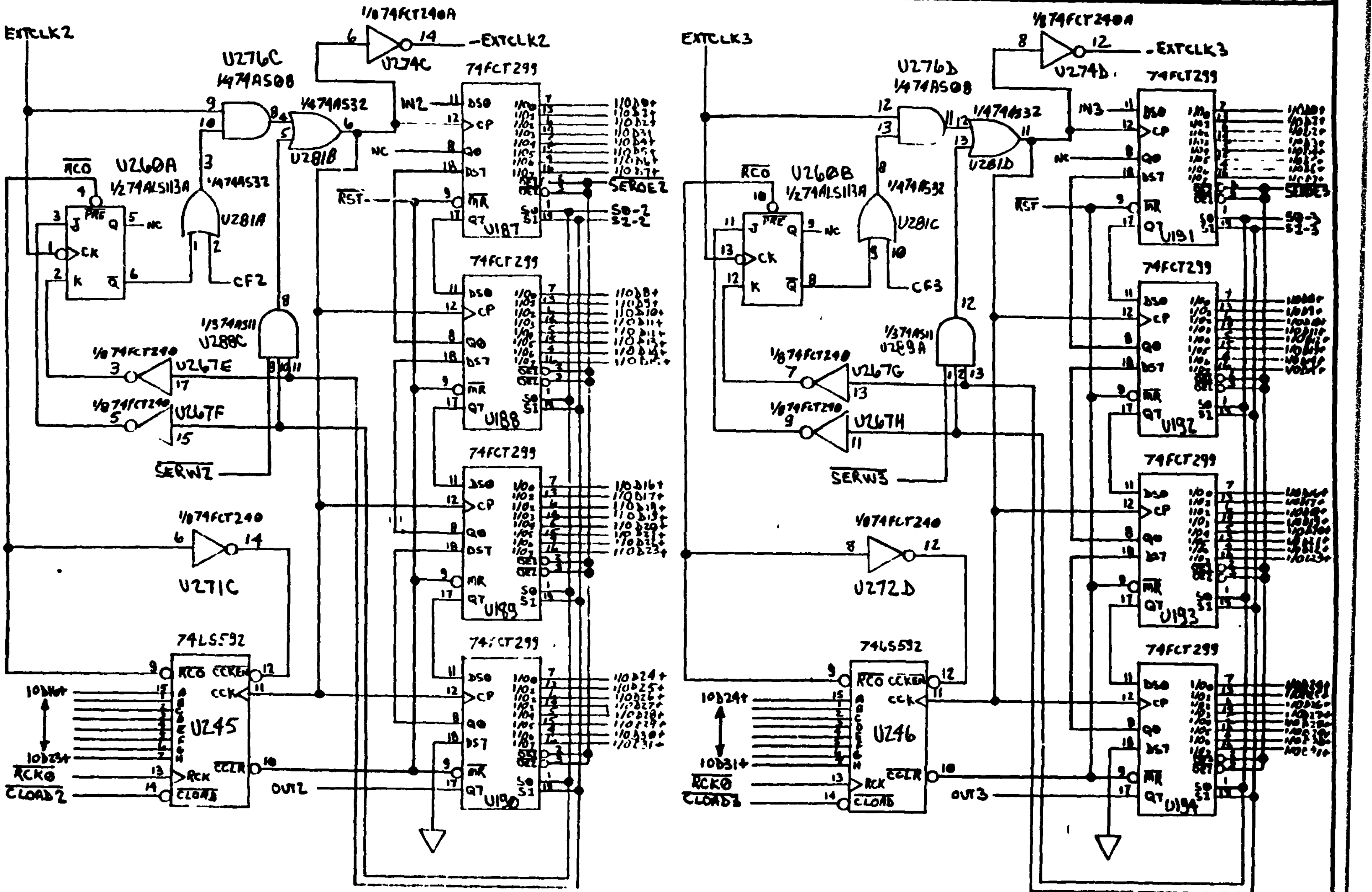
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME: SERIAL CHANNELS

IDENTIFICATION



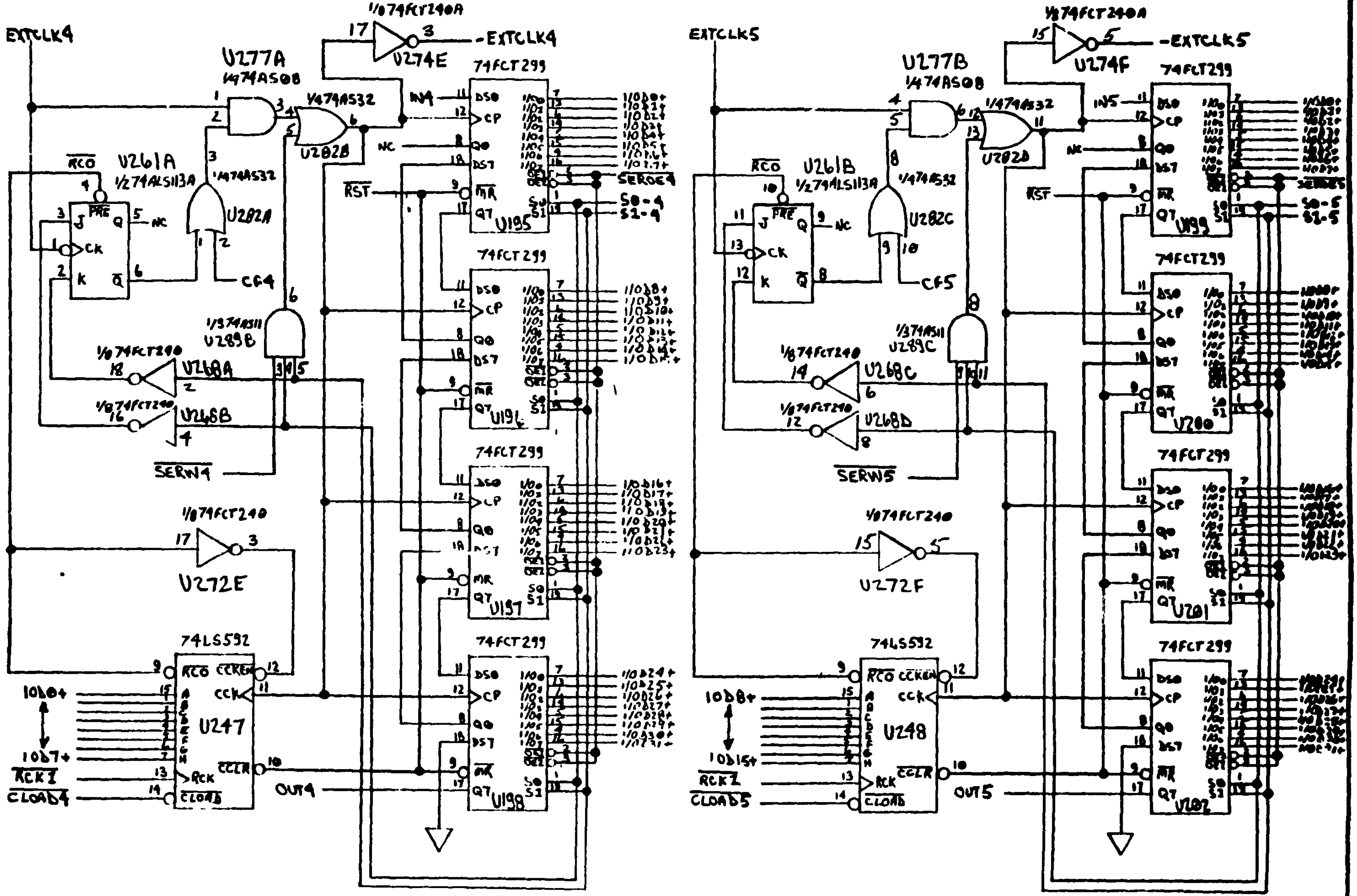
PREPARED BY – ÉTABLI PAR | DATE | PAGE 2 OF/DE | PAGES

NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME: SERIAL CHANNELS | IDENTIFICATION



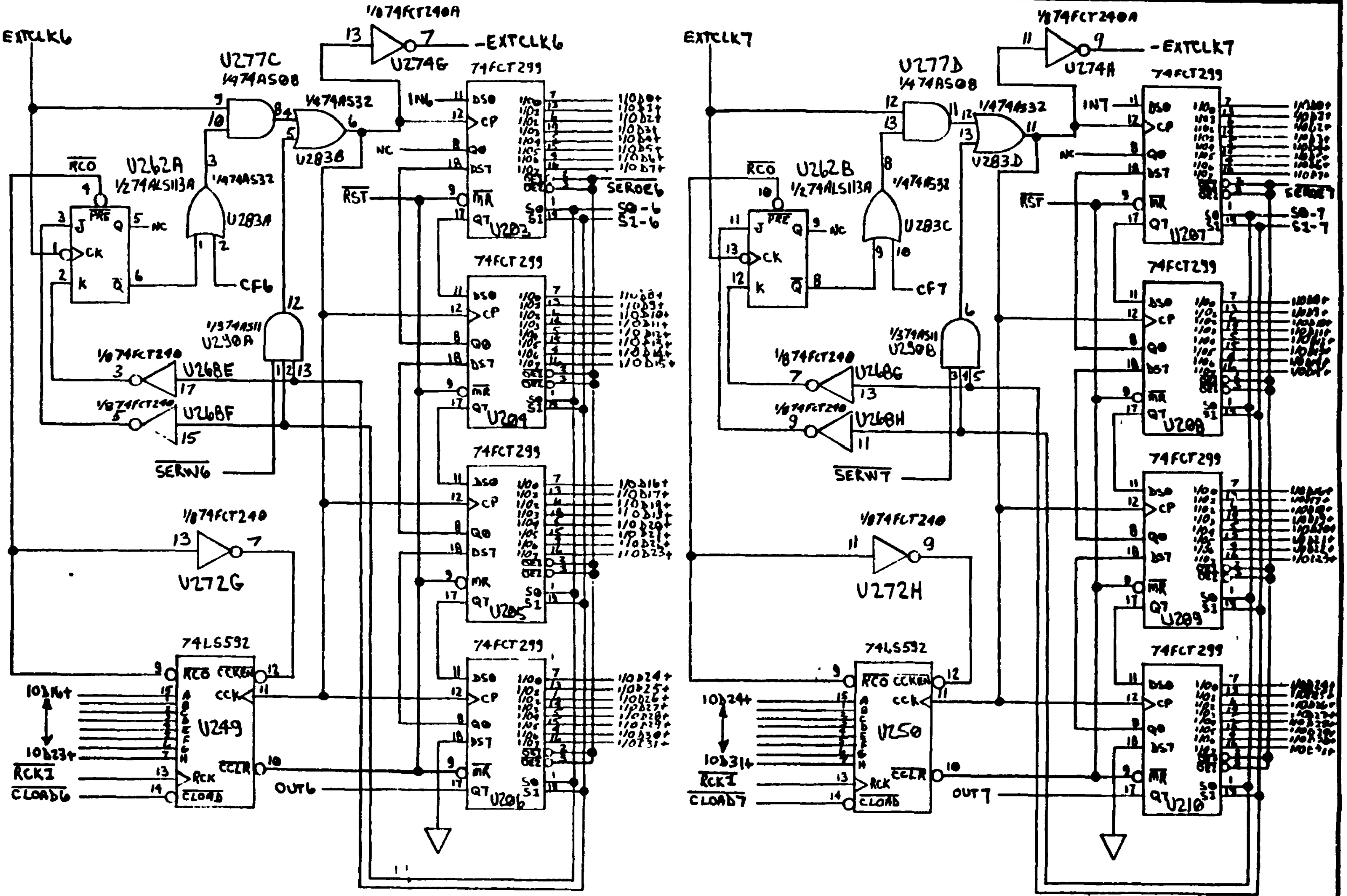
PREPARED BY – ÉTABLI PAR | DATE | PAGE 3 OF PAGES

NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME: SERIAL CHANNELS | IDENTIFICATION

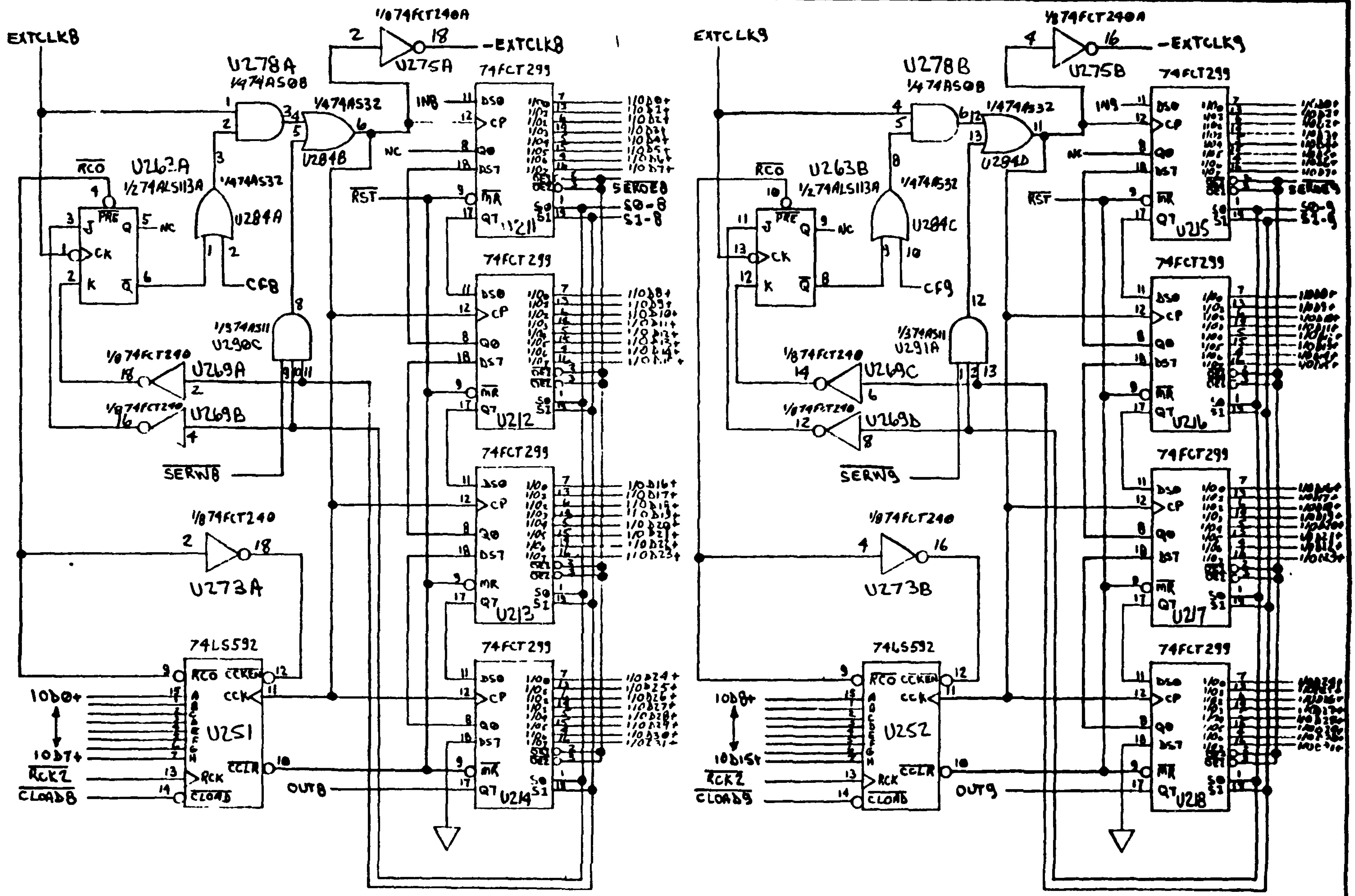


FORM 8 10 – 11 (F) E

PREPARED BY – ÉTABLI PAR	DATE	PAGE 4 OF PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME SERIAL CHANNELS		IDENTIFICATION



PREPARED BY — ÉTABLIE PAR	DATE	PAGE 5 OF PAGES
NAME OF SYSTEM OR PROGRAM — NOM DU SYSTÈME OU PROGRAMME SERIAL CHANNELS		IDENTIFICATION

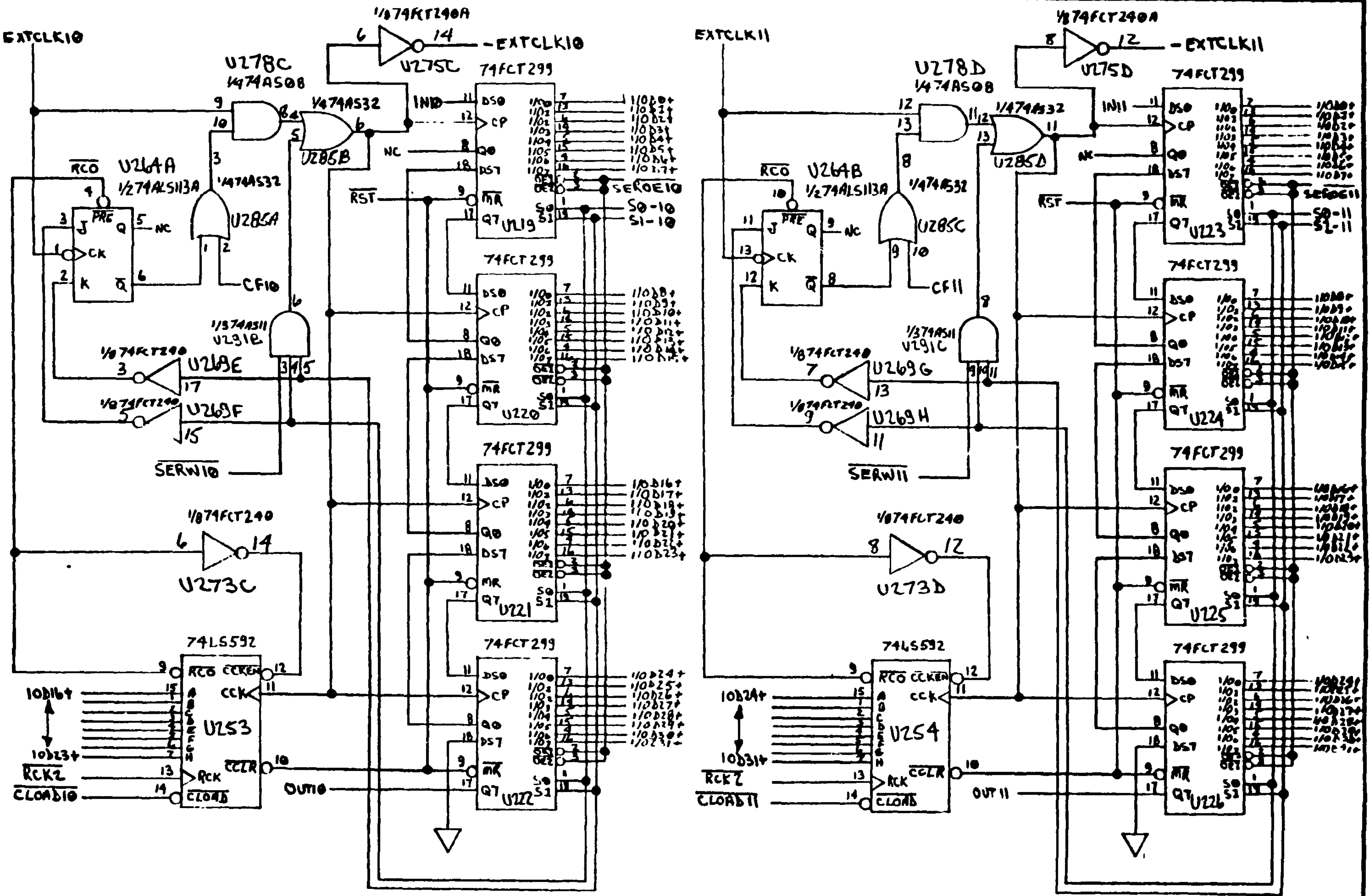


PREPARED BY – ÉTABLI PAR	DATE	PAGE	OF	PAGES
		6		

NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME

SERIAL CHANNELS

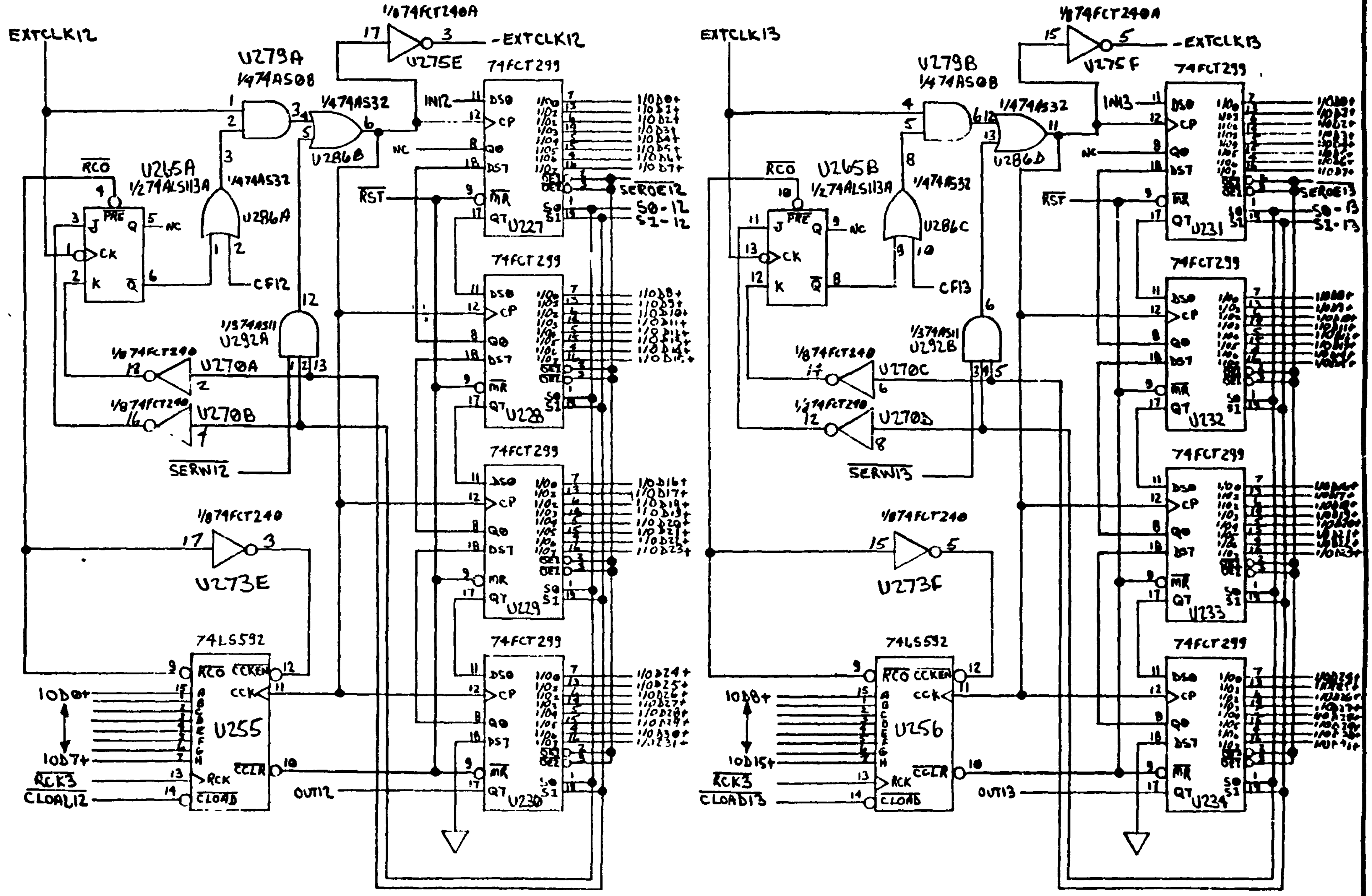
IDENTIFICATION



PREPARED BY – ÉTABLIE PAR | DATE | PAGE 7 OF | PAGES

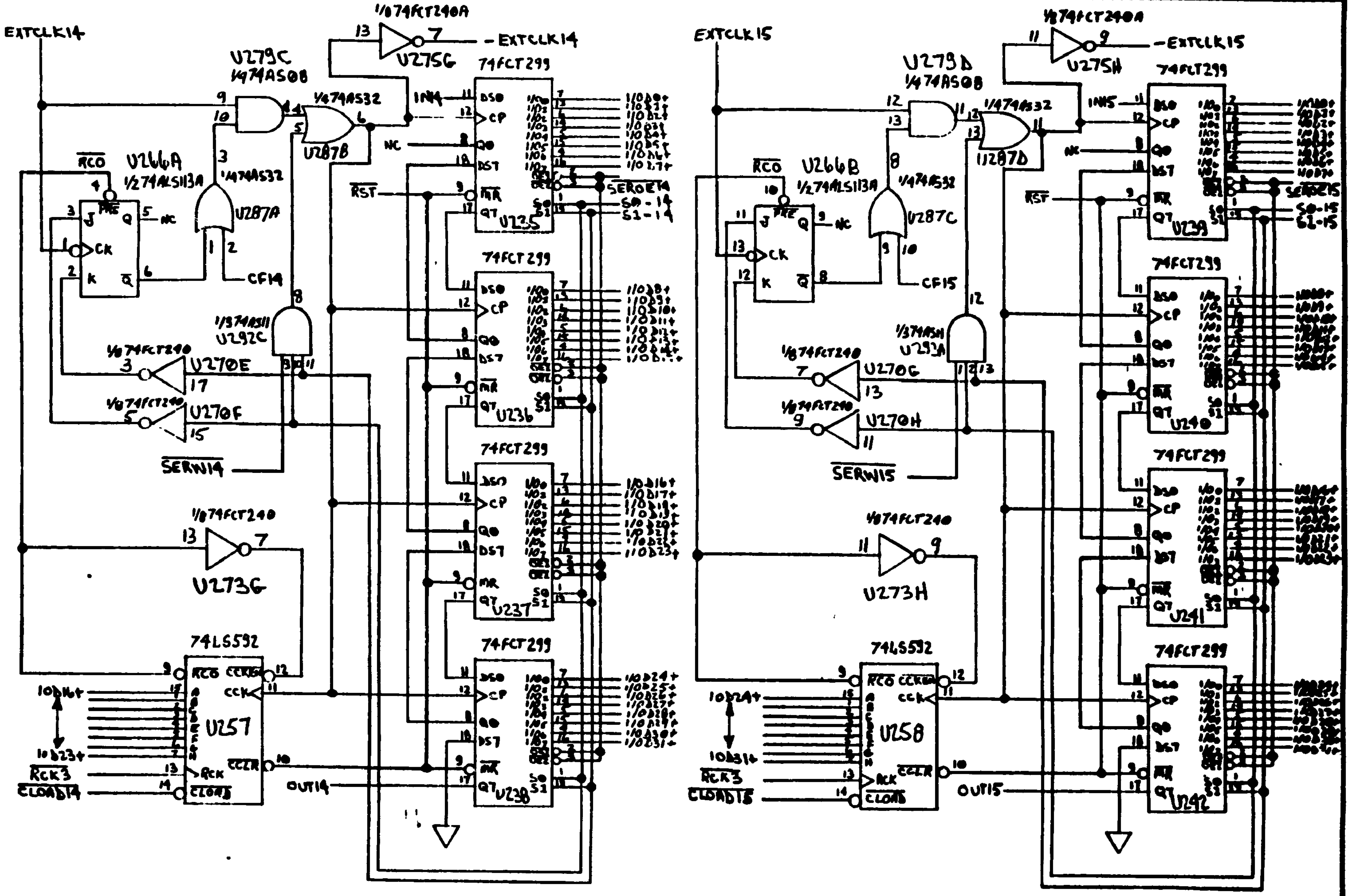
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME: SERIAL CHANNELS

IDENTIFICATION

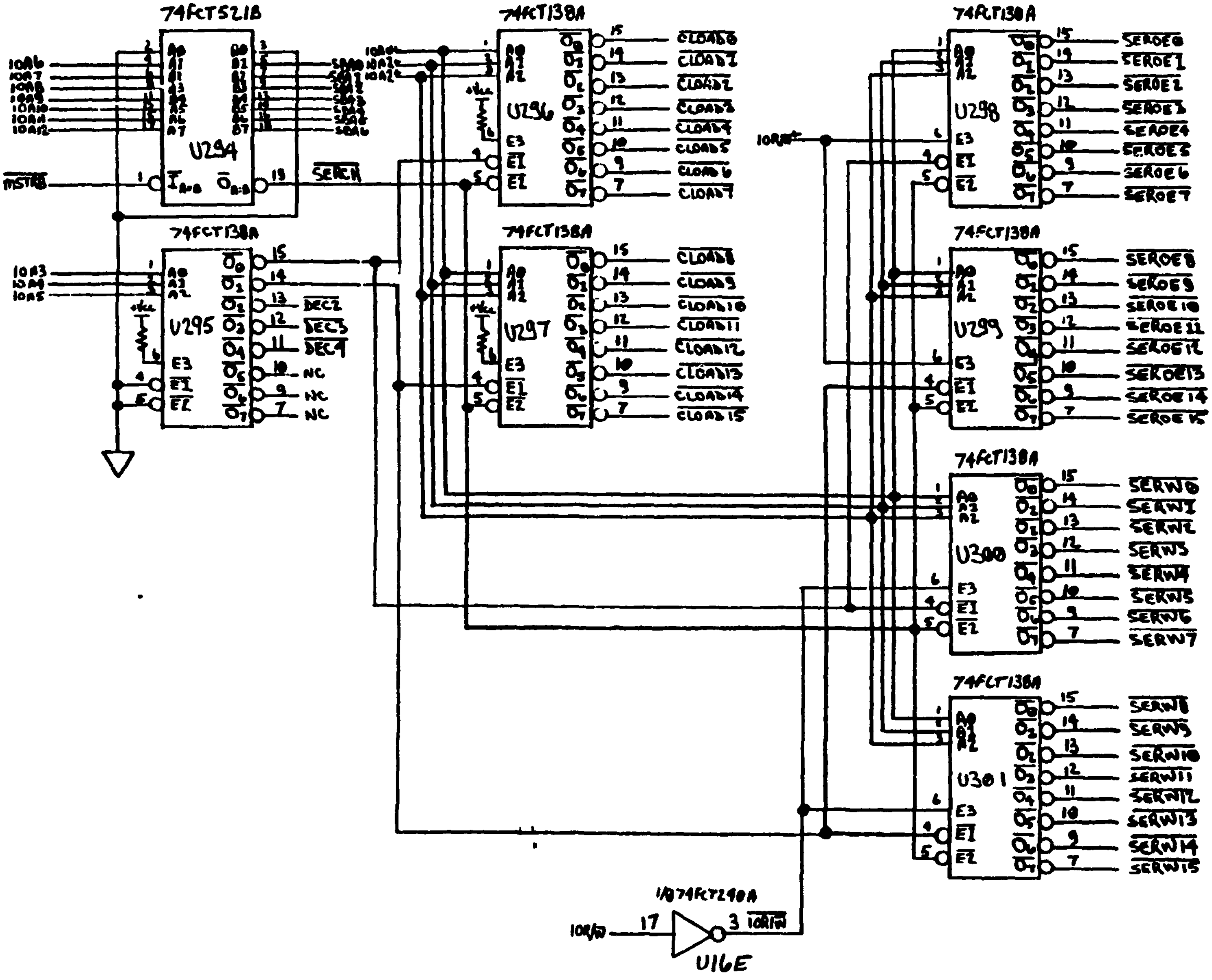


PREPARED BY – ÉTABLI PAR | DATE | PAGE 8 OF PAGES

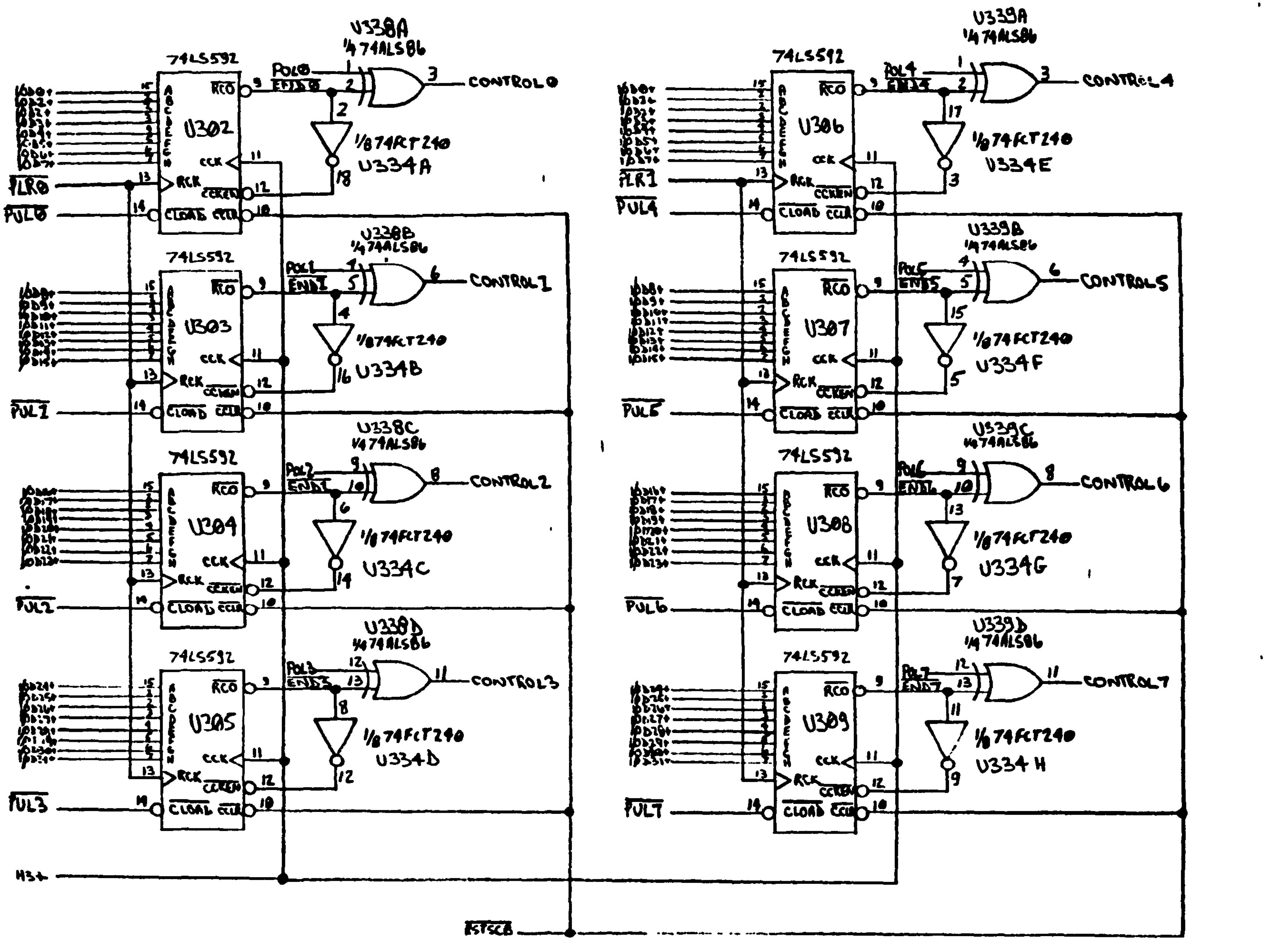
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME: SERIAL CHANNELS | IDENTIFICATION



PREPARED BY – ÉTABLI PAR	DATE	PAGE OF DE PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME SERIAL CHANNELS DECODE	IDENTIFICATION	

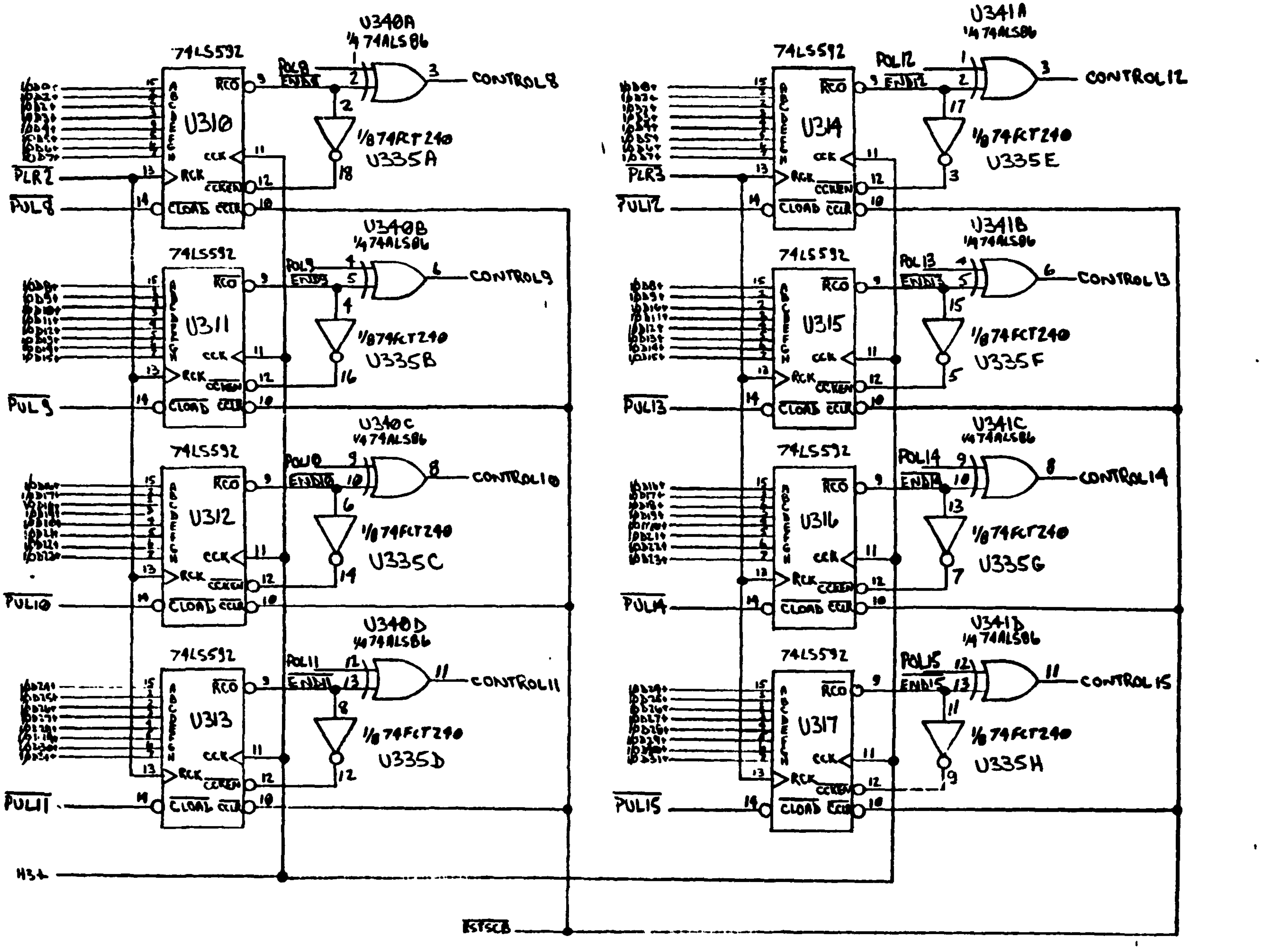


PREPARED BY – ÉTABLI PAR	DATE	PAGE 1 OF PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME SERIAL CONTROL BLOCK	IDENTIFICATION	

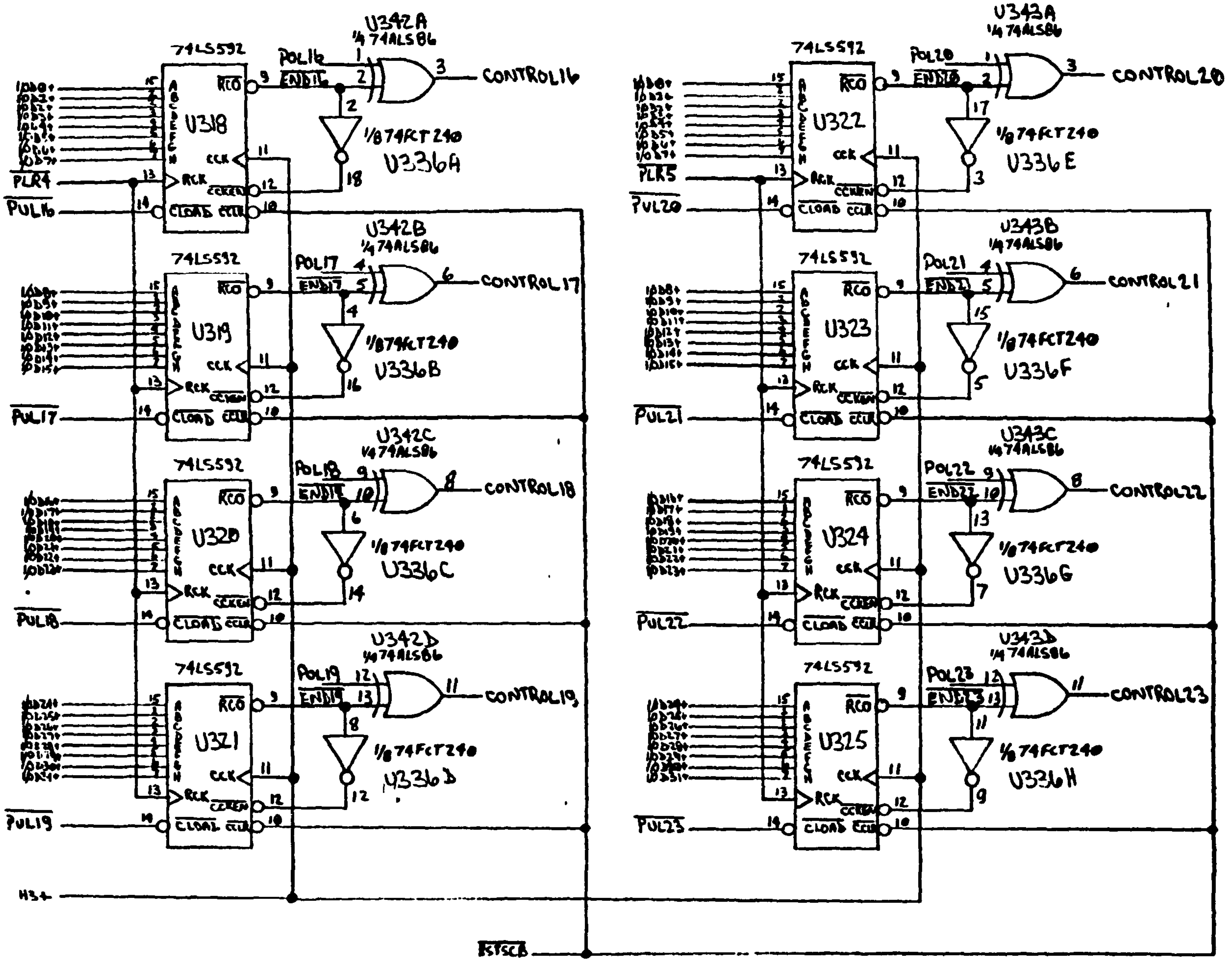


D1C

PREPARED BY – ÉTABLI PAR	DATE	PAGE 2 OF PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME SERIAL CONTROL BLOCK		IDENTIFICATION

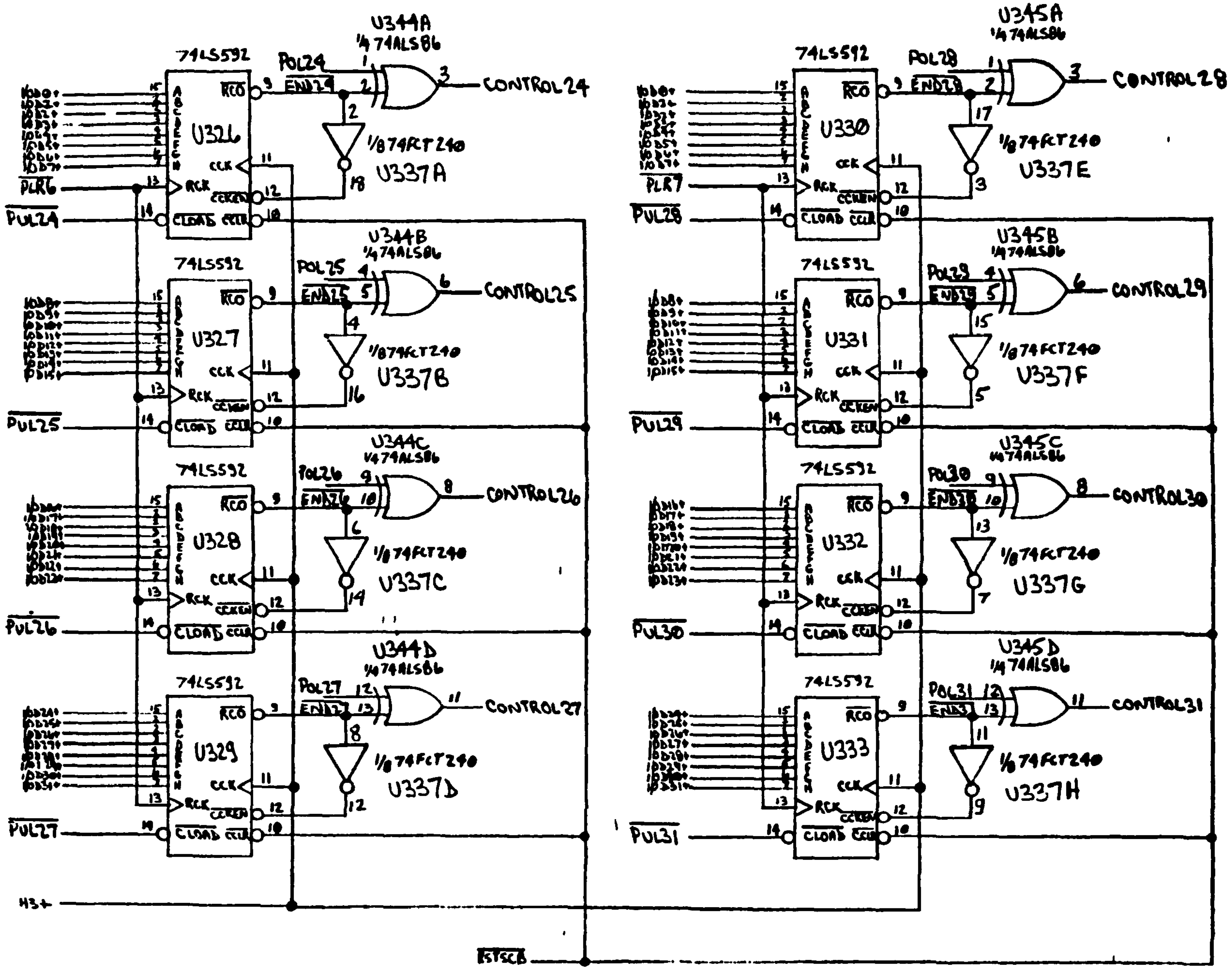


PREPARED BY – ÉTABLI PAR	DATE	PAGE 3 OF DE	PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME SERIAL CONTROL BLOCK	IDENTIFICATION		

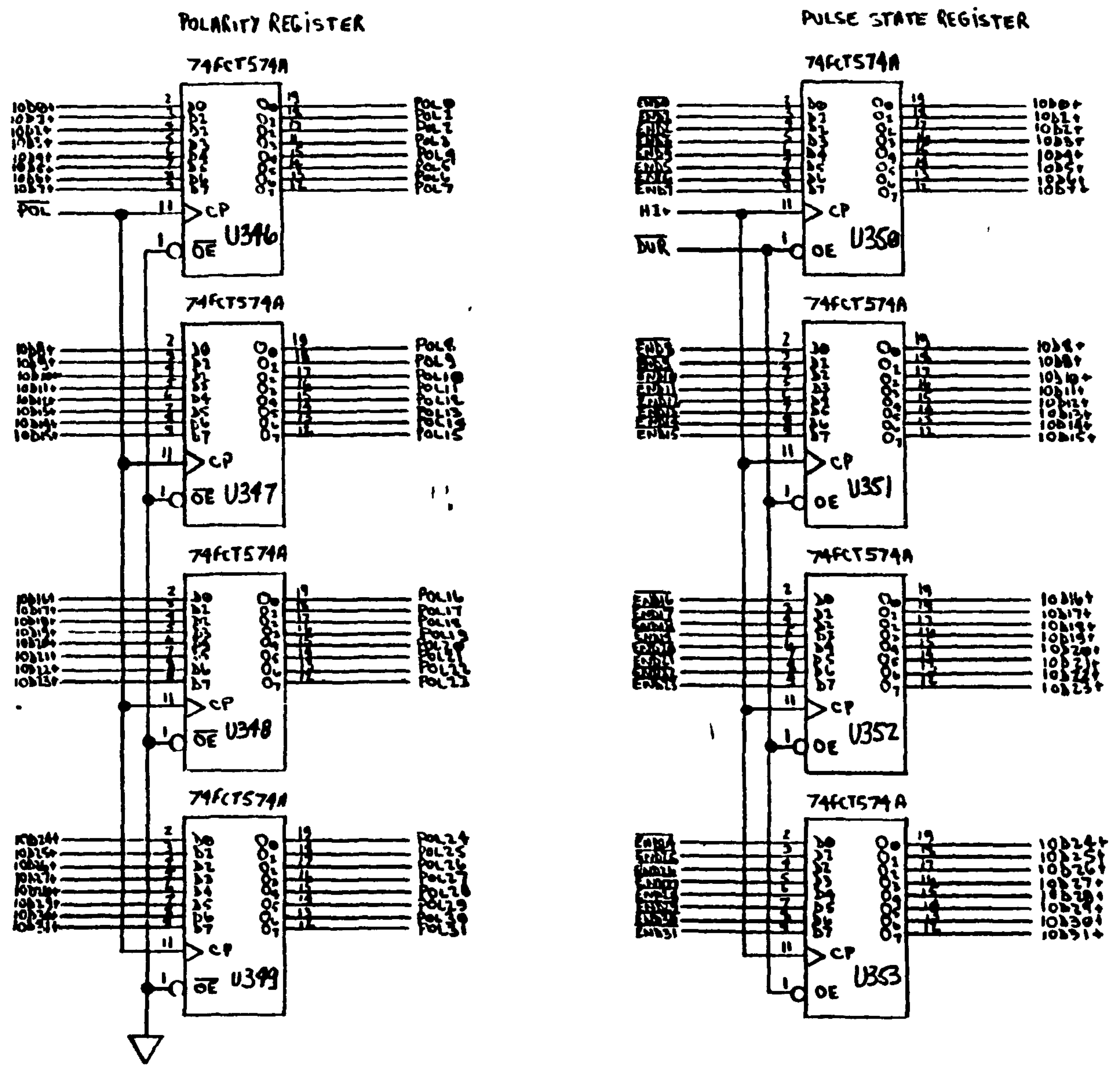


PREPARED BY – ÉTABLI PAR	DATE	PAGE	PAGES
		4 OF DE	

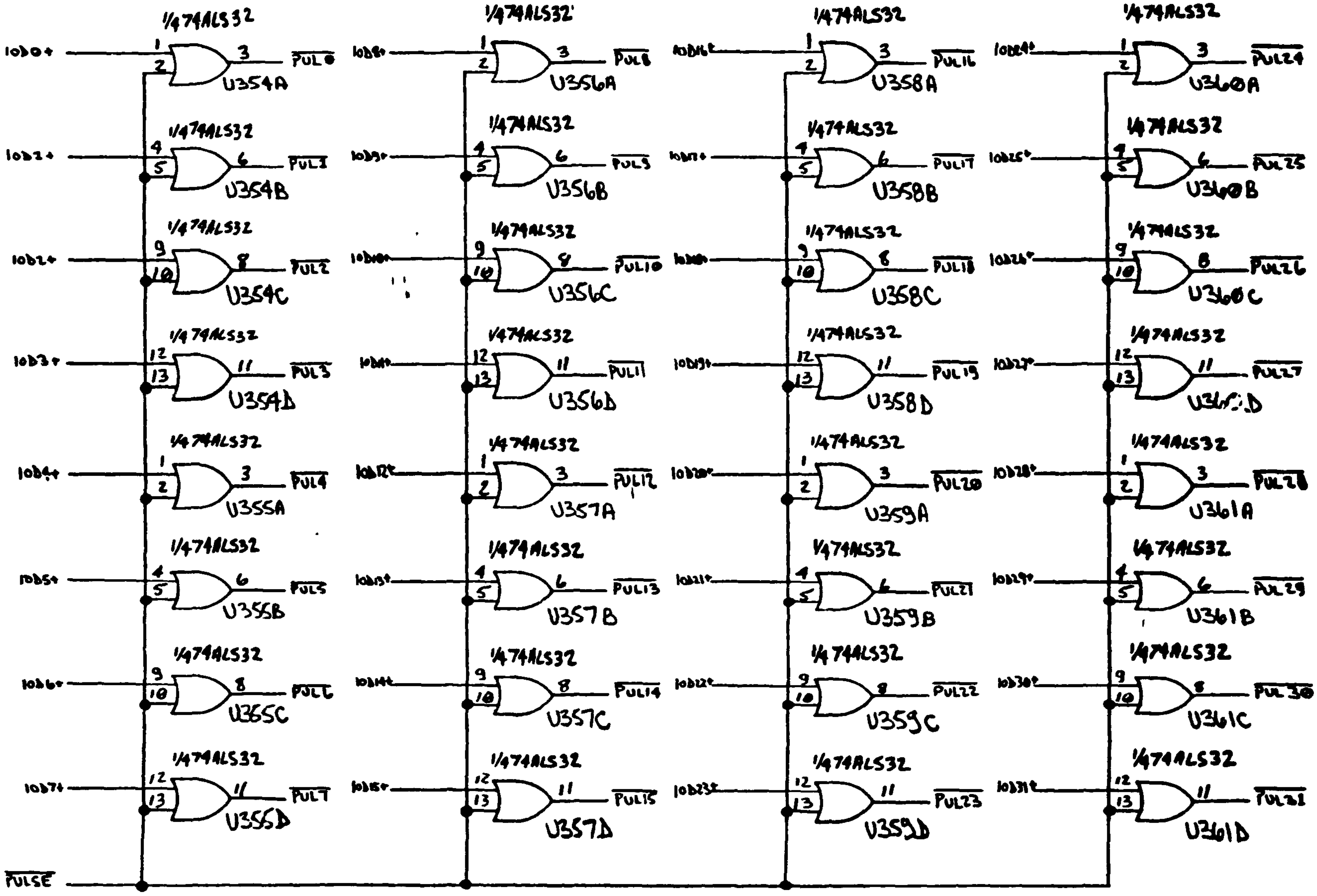
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME	IDENTIFICATION
SERIAL CONTROL BLOCK	



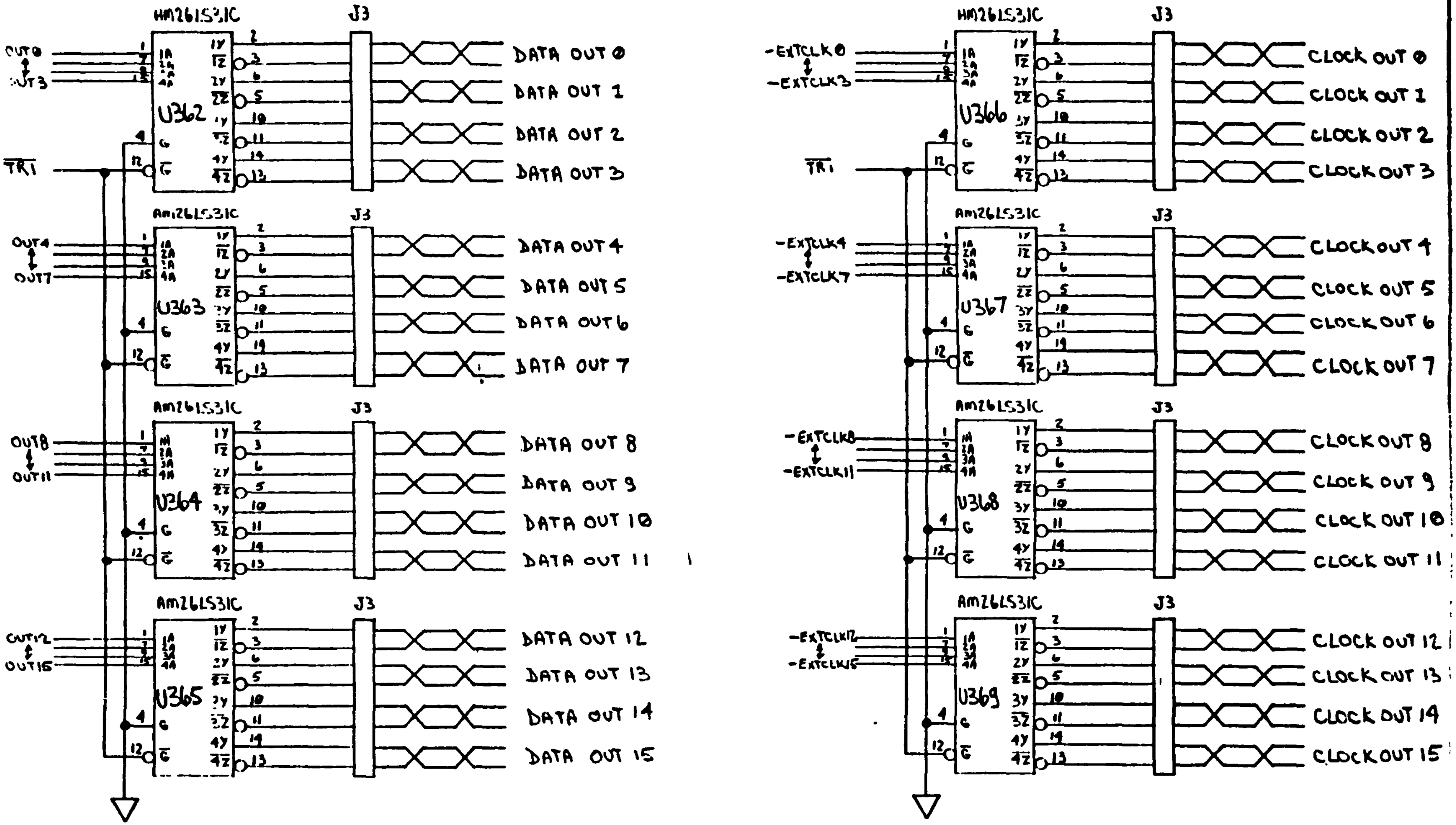
PREPARED BY – ÉTABLI PAR	DATE	PAGE OF DE PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME SERIAL CONTROL BLOCK REGISTERS	IDENTIFICATION	



PREPARED BY – ÉTABLI PAR	DATE	PAGE OF DE
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME SERIAL CONTROL BLOCK / START CONTROL PULSES	IDENTIFICATION	



PREPARED BY – ETABLIE PAR	DATE	PAGE 1 OF PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME RS422 INTERFACE		IDENTIFICATION

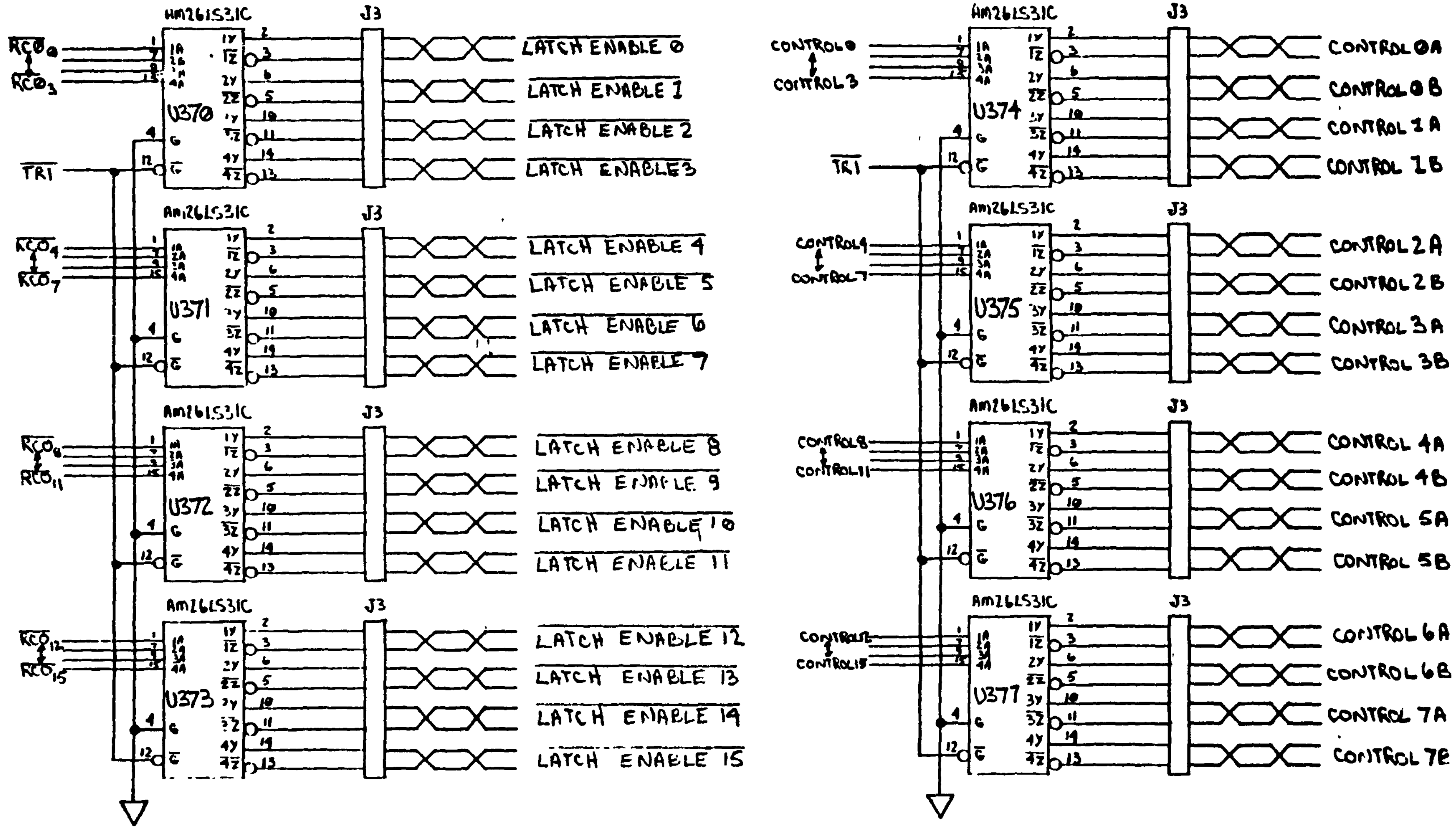


17/6

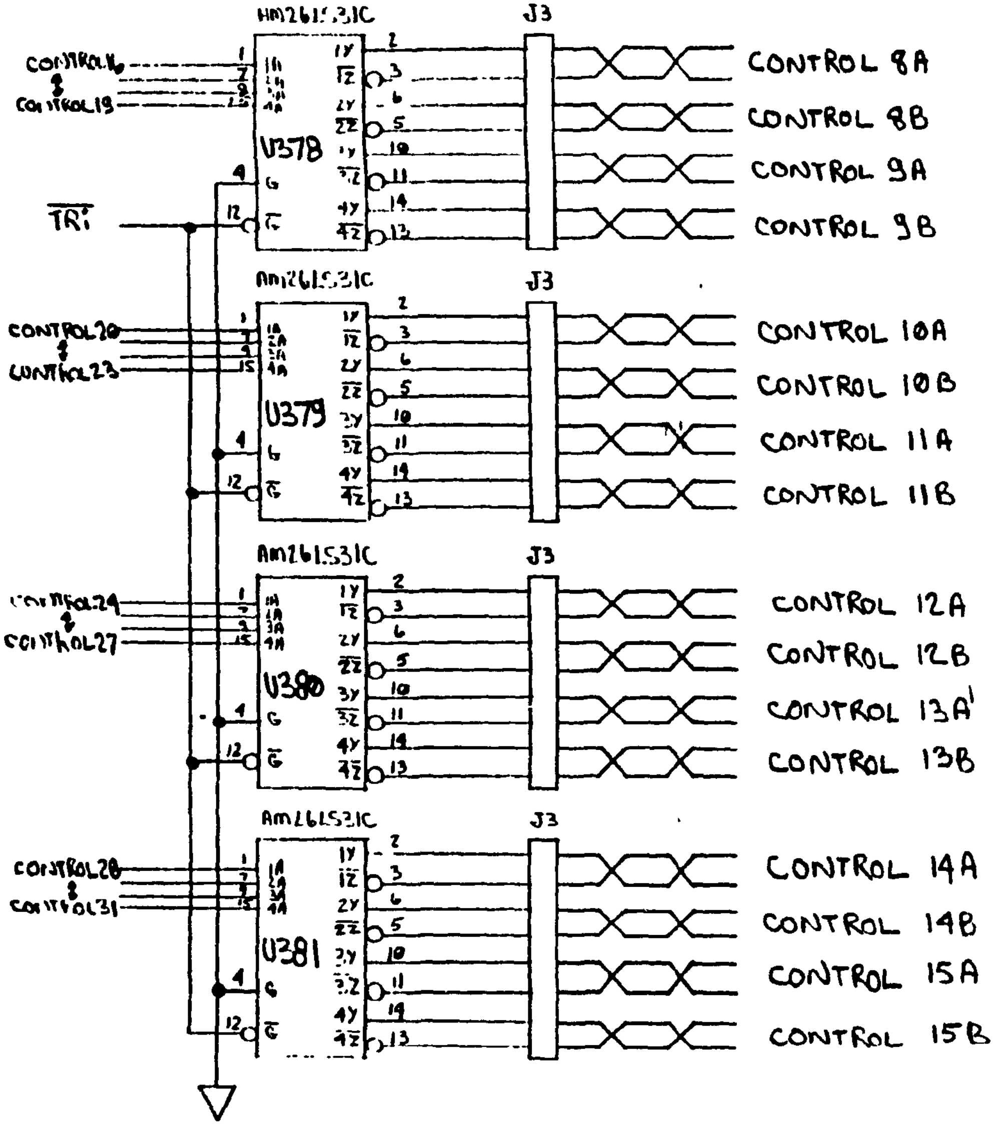
PREPARED BY – ÉTABLI PAR | DATE | PAGE 2 OF DE

NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU DU PROGRAMME: RS422 INTERFACE

IDENTIFICATION

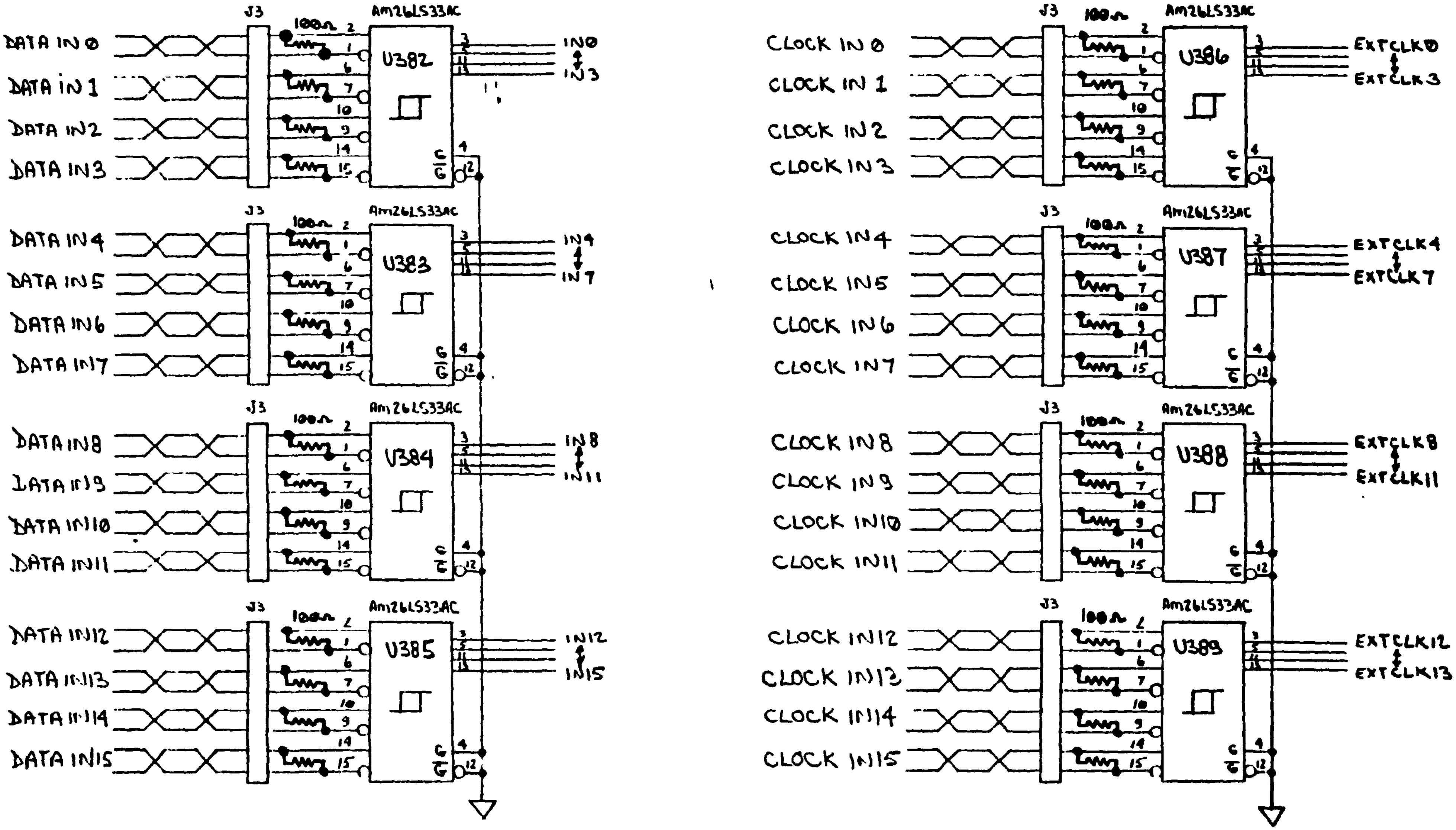


PREPARED BY – ETABLI PAR	DATE	PAGE 3 OF
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME		IDENTIFICATION
RS422 INTERFACE		



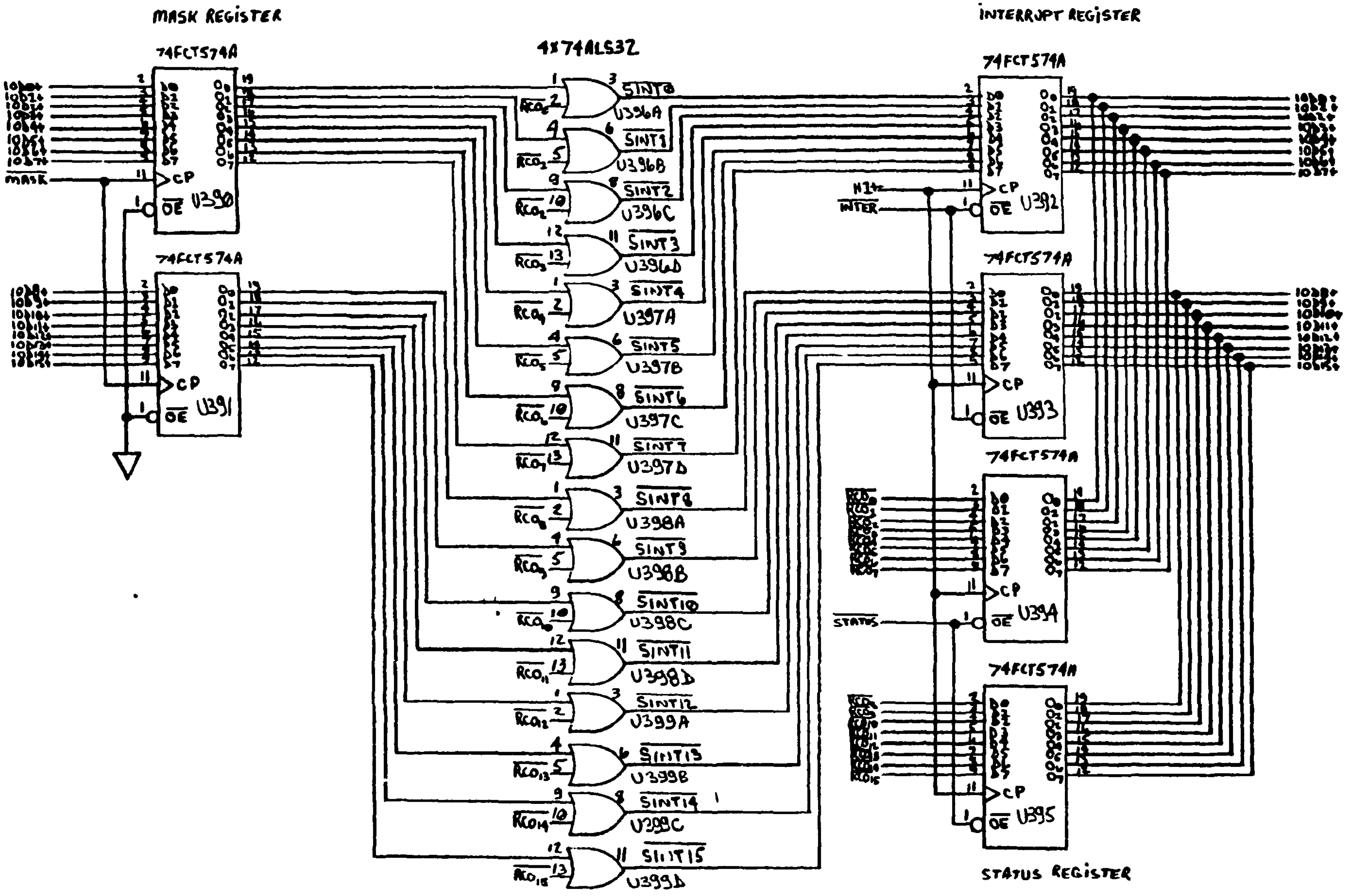
PREPARED BY – ÉTABLI PAR		PAGE	PAGES
		4 OF	

NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME	IDENTIFICATION
RS422 INTERFACE	



219

PREPARED BY – ÉTABLIE PAR	DATE	PAGE OF DE PAGES
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME INTERRUPT CIRCUITRY	IDENTIFICATION	



APPENDIX E

INTERFACE BOARD

Sheet E1 - Initialization program preload circuitry (also E2)

Sheet E2 - DRQ3B bus receivers and input two-wire handshaking circuitry

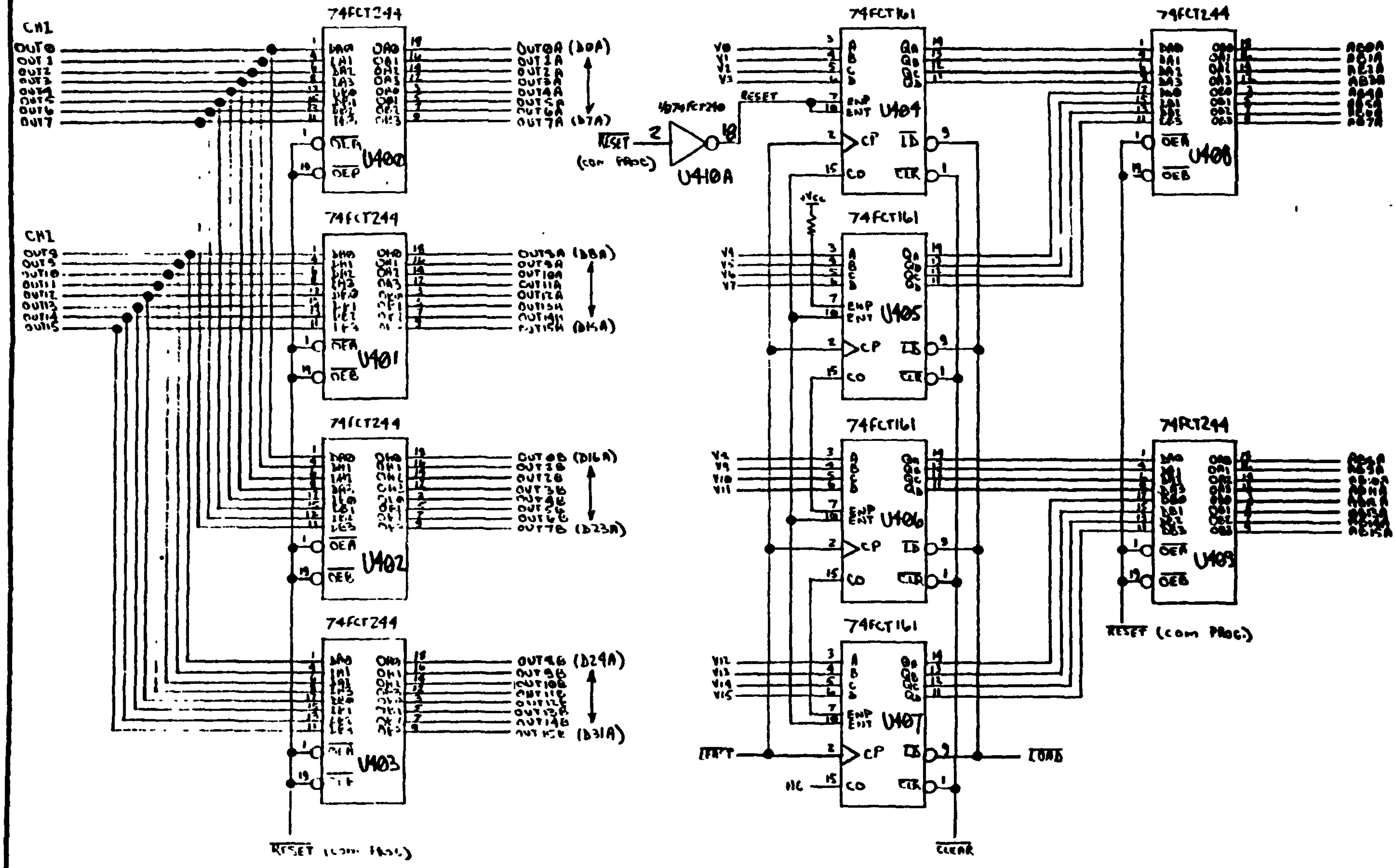
Sheet E3 - DRQ3B bus drivers and output two-wire handshaking circuitry

Sheet E4 - Function registers (also E3), and miscellaneous

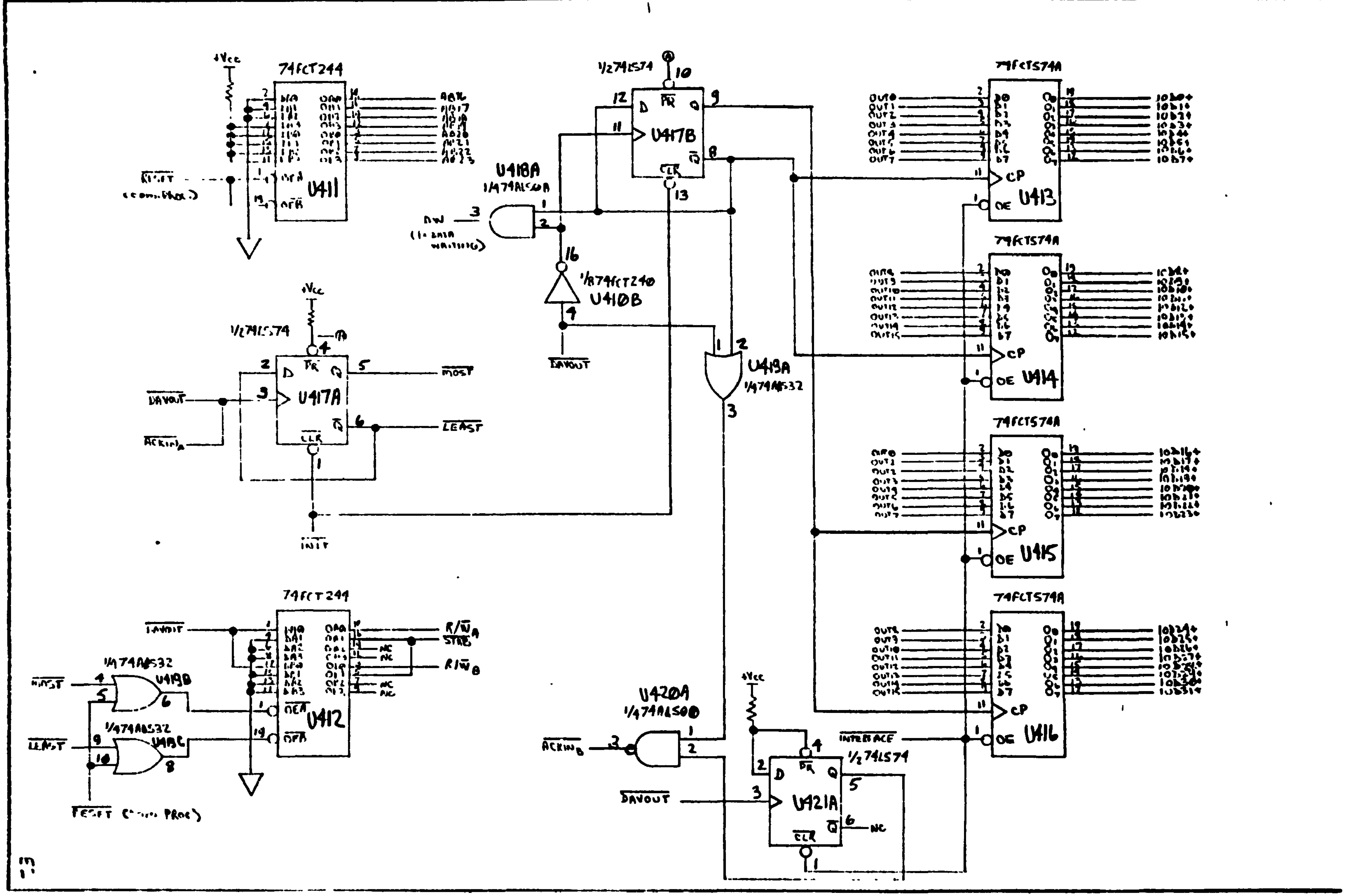
Interface board

Type	Description	Quantity
74FCT244	Fast CMOS octal buffer/line driver	8
74FCT161	Fast CMOS synchronous presettable binary counter	4
74FCT240	Fast CMOS octal buffer/line driver	1
74FCT574A	Fast CMOS octal D register (3-state)	11
74LS74	Dual D-type positive-edge-triggered flip-flops	3
74ALS08	Quadruple 2-input positive-AND gate	1
74F32	Quadruple 2-input positive-OR gates	2
74F00	Quadruple 2-input positive-NAND gates	1
74F27	Triple 3-input positive-NOR gates	1
74F08	Quadruple 2-input positive-AND gates	1
Total	33	

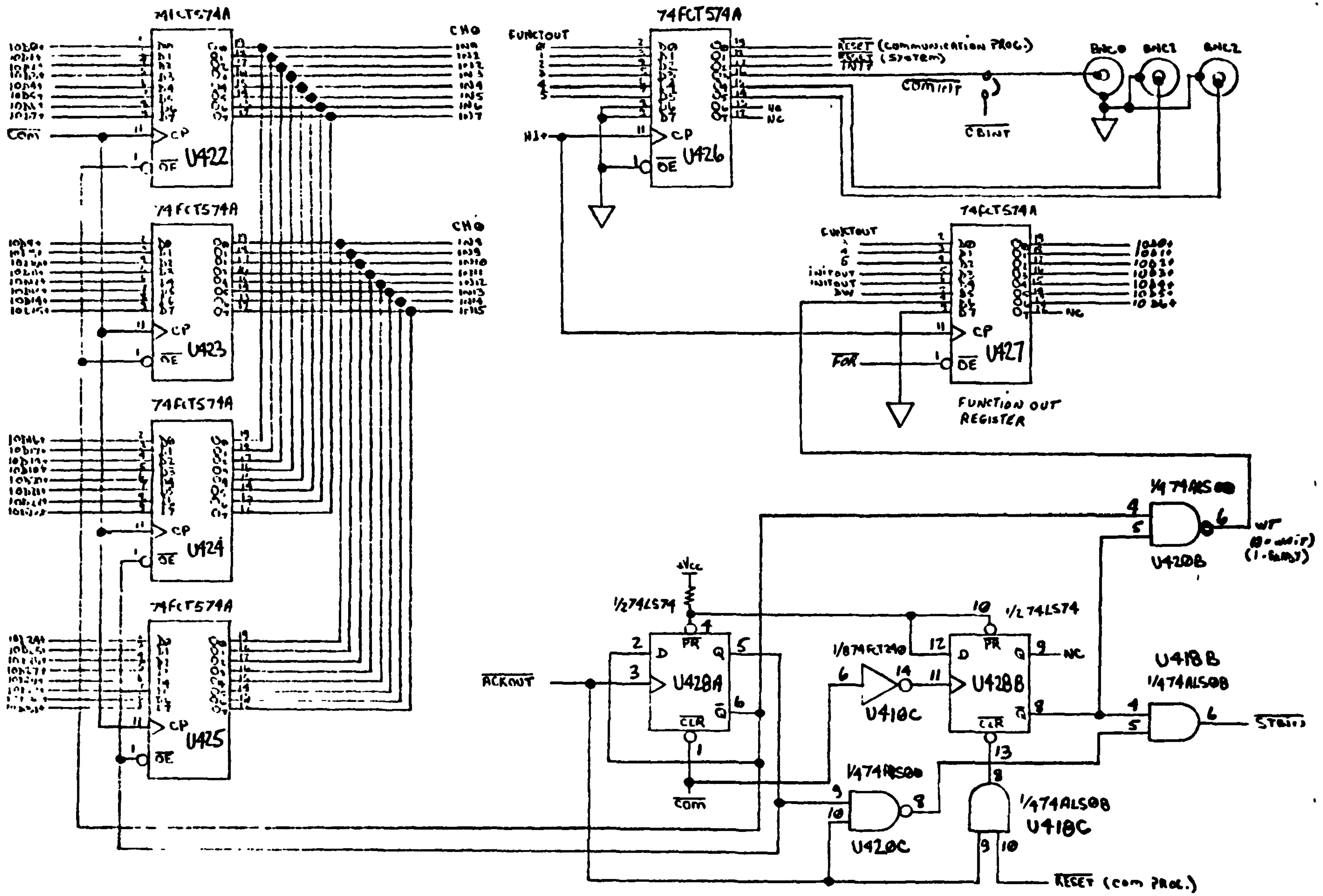
PREPARED BY – ÉTABLI PAR	DATE	PAGE 1 OF
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME INTERFACE BOARD		IDENTIFICATION



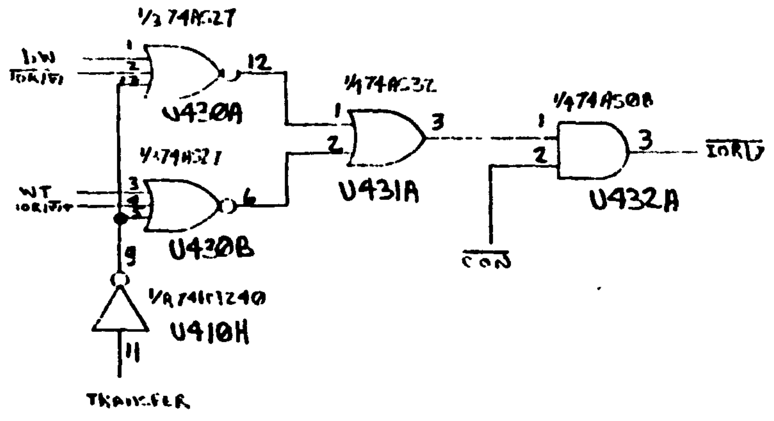
PREPARED BY — ÉTABLIE PAR	DATE	PAGE 2 OF
NAME OF SYSTEM OR PROGRAM — NOM DU SYSTÈME OU PROGRAMME INTERFACE BOARD	IDENTIFICATION	



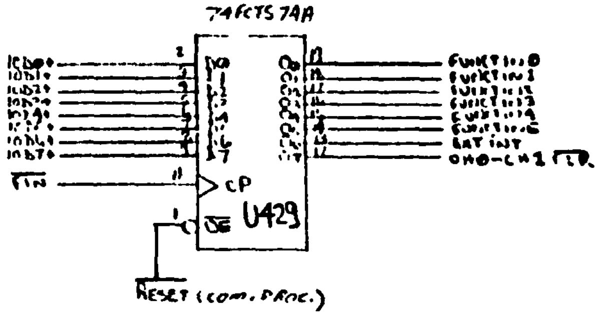
PREPARED BY – ÉTABLI PAR	DATE	PAGE 3 OF
NAME OF SYSTEM OR PROGRAM – NOM DU SYSTÈME OU PROGRAMME	IDENTIFICATION	
INTERFACE BOARD		

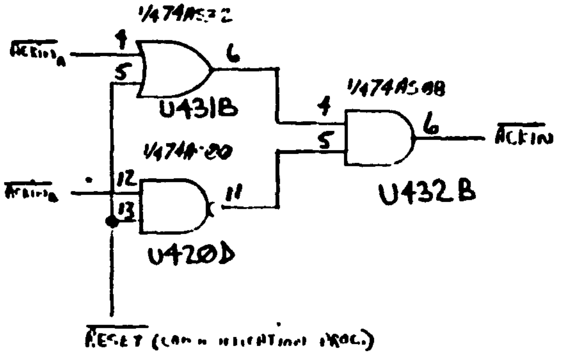


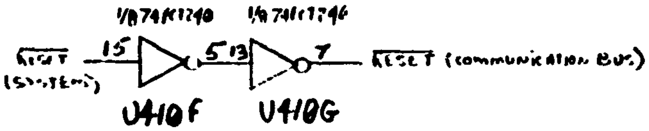
PREPARED BY - ÉTABLI PAR	DATE	PAGE 4 OF DE
NAME OF SYSTEM OR PROGRAM - NOM DU SYSTÈME OU PROGRAMME INTERFACE BOARD	IDENTIFICATION	



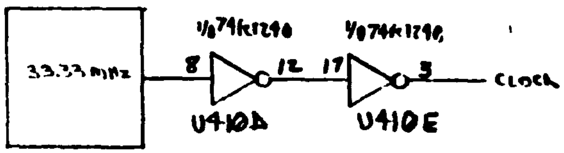
FUNCTION IN REGISTER







CLOCK GENERATION

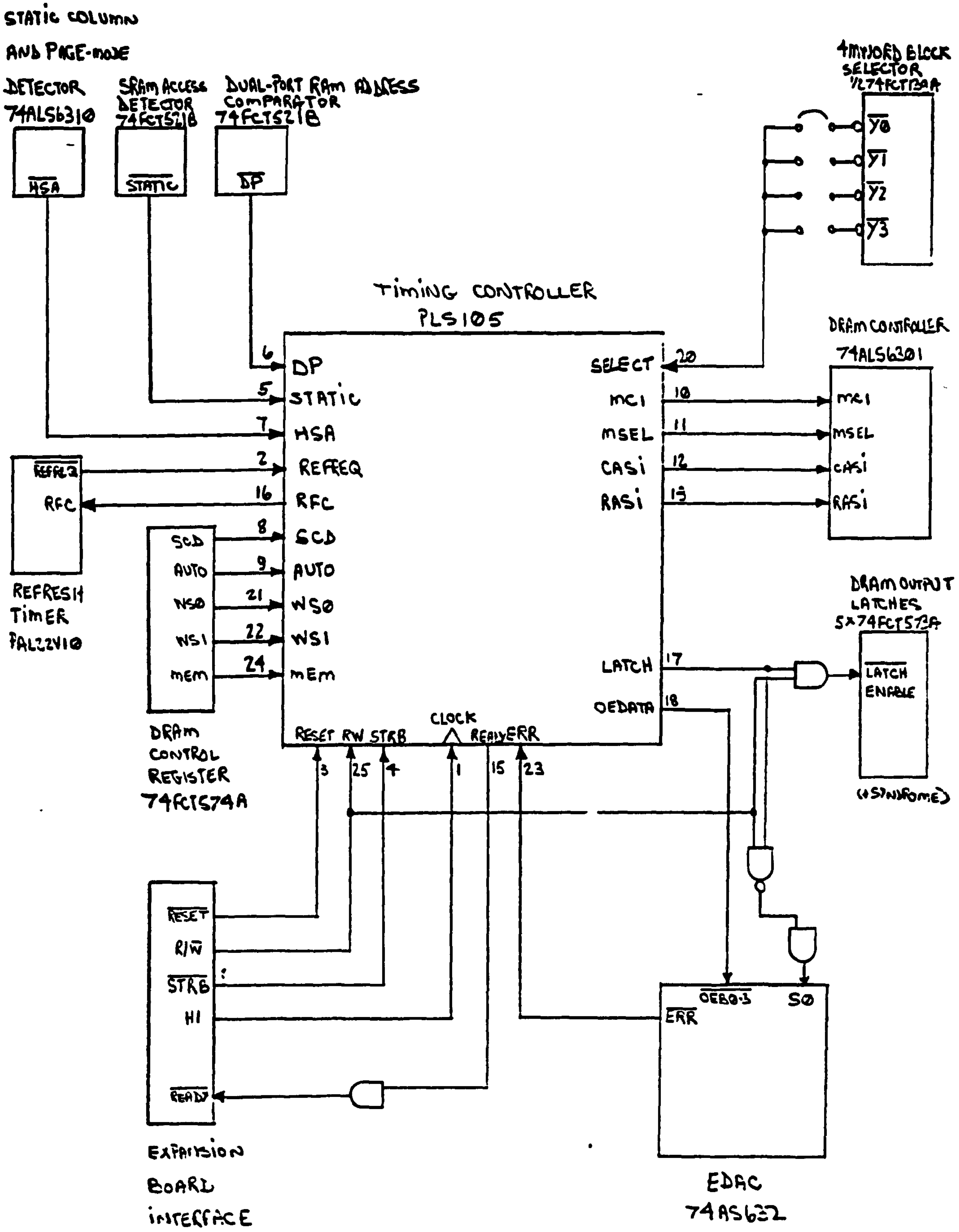


APPENDIX F

DRAM INTERFACE

Sheet F1 - Timing controller (basic schematic)
Sheet F2 - Description of the inputs of the timing controller
Sheet F3 - Description of the outputs. string declaration. and state header
Sheet F4 - Outputs vs states. and state descriptions
Sheet F5 - State and transition definitions
Sheet F6 - Branch conditions
Sheet F7 - Inter-state conditions
Sheet F8 - State diagram for the timing controller
Sheet F9 - DRAM controller (basic schematic)
Sheet F10 - Memory driver (basic schematic)
Sheet F11 - Bus/EDAC controller (basic schematic)
Sheet F12 - EDAC (basic schematic)
Sheet F13 - Refresh timer (basic schematic)
Sheet F14 - PAL equations for the refresh timer
Sheet F15 - Simulation for the refresh timer
Sheet F16 - Fuse plot for the refresh timer
Sheet F17 - Fuse plot for the refresh timer
Sheet F18 - Fuse plot for the refresh timer
Sheet F19 - JEDEC file for the refresh timer
Sheet F20 - JEDEC file for the refresh timer (test pattern)
Sheet F21 - Simulation selective trace listing for the refresh timer
Sheet F22 - Simulation history listing for the refresh timer

TIMING CONTROLLER



```
ITLE      Timing Controller
ATTERN    TIMING.PDS
EVISION   1
UTHOR     S. Martel
OMPANY    McGill University
E         9 August 1988
          TIMING_CNTRL  PLS105

;*** PINS ***

; CLK    IN     IN     IN     IN     IN     IN     IN     IN     OUT    OUT    OUT
; 1      2      3      4      5      6      7      8      9      10     11     12
 CLOCK REFREQ RESET STRB  STATIC DP     HSA    SCD    AUTO   MC1    MSEL   CASI

; OUT    GND    OUT    OUT    OUT    OUT    P/E    IN     IN     IN     IN     IN
; 13     14     15     16     17     18     19     20     21     22     23     24
  RASI   GND    READY RFC    LATCH  OEDATA GND    SELECT WS0    WS1    ERR    MEM

; IN     IN     IN     VCC
; 25     26     27     28
  RW     NC     NC     VCC

  P0 P1 P2 P3 P4 P5 COMP

;*** INPUTS ***

;CLOCK   : Clock H1 provided by the processor with a frequency of 16.5 MHz.
;REFREQ  : Refresh request from the refresh timer. When REFREQ is low, the
;          timing controller must complete the DRAM access if any, and refresh
;          the next row. RAS only refresh cycle is implemented.
;RESET   : System reset (active low).
;STRB    : Local memory access strobe is low when the DRAMs, the SRAMs, or
;          dual-port RAMs or anythings are accessed on the memory bus.
;STATIC  : SRAM access strobe active low. If STATIC is low, the DRAMs shall
;          not be accessed.
;DP      : Dual-port RAM access strobe active low. When DP is low, the DRAMs
;          shall not be accessed.
;HSA     : High speed access. This is used with the static column decode
;          access mode. When the row section of the DRAM address is the same
;          as the previous DRAM address, new data can be accessed by simply
;          changing the column addresses. This reduces the access time to
;          the DRAMs. When HSA is low, high-speed accesses can be performed.
;          HSA is provided by the static column and page-mode detector.
;SCD     : Static column decode switch. SCD when high, allows static column
;          decode access otherwise high speed access is disable. AUTO must
;          be low.
;AUTO    : Automatic feature that allows the system to switch between normal
;          accesses and static column decode accesses without the user's
;          intervention. AUTO is active high and overides SCD.
;SELECT  : Up to 16 Mwords of local memory can be addressed by the processor.
;          The system is designed in order to have up to 4 Mwords of relatively
;          fast DRAMs and an optional extension of up to 12 Mwords involving
;          the same or a different number of wait-states when accessed. SELECT
;          indicates to the timing controller which part is accessed. SELECT
;          is low when the 4 Mwords part is accessed, high otherwise.
;WS0(1)  : Indicates the number of extra wait-states (relative to the 4 Mwords
;          DRAM part) that must be add when the extension (DRAMs) is accessed.
;          WS0,WS1 -> 00 : + 0 wait-state (same access time as 4 Mwords)
;                  -> 10 : + 1 wait-state (60 ns)
;                  -> 01 : + 2 wait-states (120 ns)
;                  -> 11 : + 3 wait-states (180 ns)
;ERR     : Error flag from the EDAC. When ERR is low, the timing controller
;          does not allow more access to the DRAMs until the correct data
;         is read from the EDAC.
;MEM    : MEM when high indicates to the timing controller that the extension
;         is used as memory. If a co-processor or anything else is assigned
;         to the extension, MEM should be set low.
;RW     : Read and write line from the processor. RW is low when a write
;         cycle is performed, RW is high otherwise.

;*** OUTPUTS ***

;MC1    : MC1 is connected to the DRAM controller. MC1 is low when refresh
;         without scrubbing must be performed. Otherwise MC1 is high-including
;         read/write cycles.
;MSEL   : MSEL is connected to the DRAM controller. MSEL controls the address
;         multiplexer in the DRAM controller and selects the row or column
;         address. When MSEL is low, the row address is sent to the DRAMs,
;         otherwise the column address is sent.
;CASI   : CASI is connected to the DRAM controller. CASI is the column address
;         strobe input.
;RASI   : RASI is connected to the DRAM controller. RASI is the row address
;         strobe input.
;READY  : READY is connected to the processor via an AND gate. During a
;         DRAM access, the processor will wait until READY is low.
;RFC    : RFC is connected to the refresh timer. The timing controller
;         sets RFC high to indicate to the refresh timer that the refresh
;         cycle is complete.
;LATCH  : LATCH is connected to the DRAM output latches. New outputs are
;         latched when LATCH is high and RW is high (RW is tested externally
;         with an AND gate). This garantees that the DRAM output is available
;         for the whole read cycle because of premature DRAM turn off to
;         minimize the delay between accesses.
;OEDATA : OEDATA is connected to the EDAC. OEDATA when low enables the correct
;         data output to the memory bus. OEDATA is active (low) when an
;         error has occured (ERR low) and during the first successive read
;         cycle to the 4 Mwords DRAM part or to the extension if used as
;         extended memory.

;*** STRING DECLARATION ***

;LOCAL_MEMORY_ACCESS defines any accesses to the memory bus excluding
;the external SRAM block and the dual-port RAMs.
STRING LOCAL_MEMORY_ACCESS ' /STRB * DP * STATIC * REFREQ * RESET '

;*** STATE HEADER ***

STATE                   ;Start the state machine section
MOORE_MACHINE           ;Does not consider the inputs as MEALY does
MASTER_RESET            ;Programmable pin selects preset
;*** DEFAULT VALUES ***

DEFAULT_BRANCH ST0  ;Initialization at state 0
DEFAULT_OUTPUT MC1 MSEL CASI RASI READY /RFC /LATCH OEDATA

;*** OUTPUTS VS STATES ***

;STATES/OUTPUTS -> MC1    MSEL    RASI    CASI    READY  RFC    LATCH  OEDATA

;ST0               1      0       1       1       1      0      0      1
                   1      0       0       1       1      0      0      1
;ST2               1      1       0       0       0      0      1      1
;ST3               1      0       1       1       1      0      0      1
;ST4               0      0       0       1       1      1      0      1
;ST5               0      0       0       1       1      1      0      1
;ST6               1      0       1       1       1      0      0      1
```

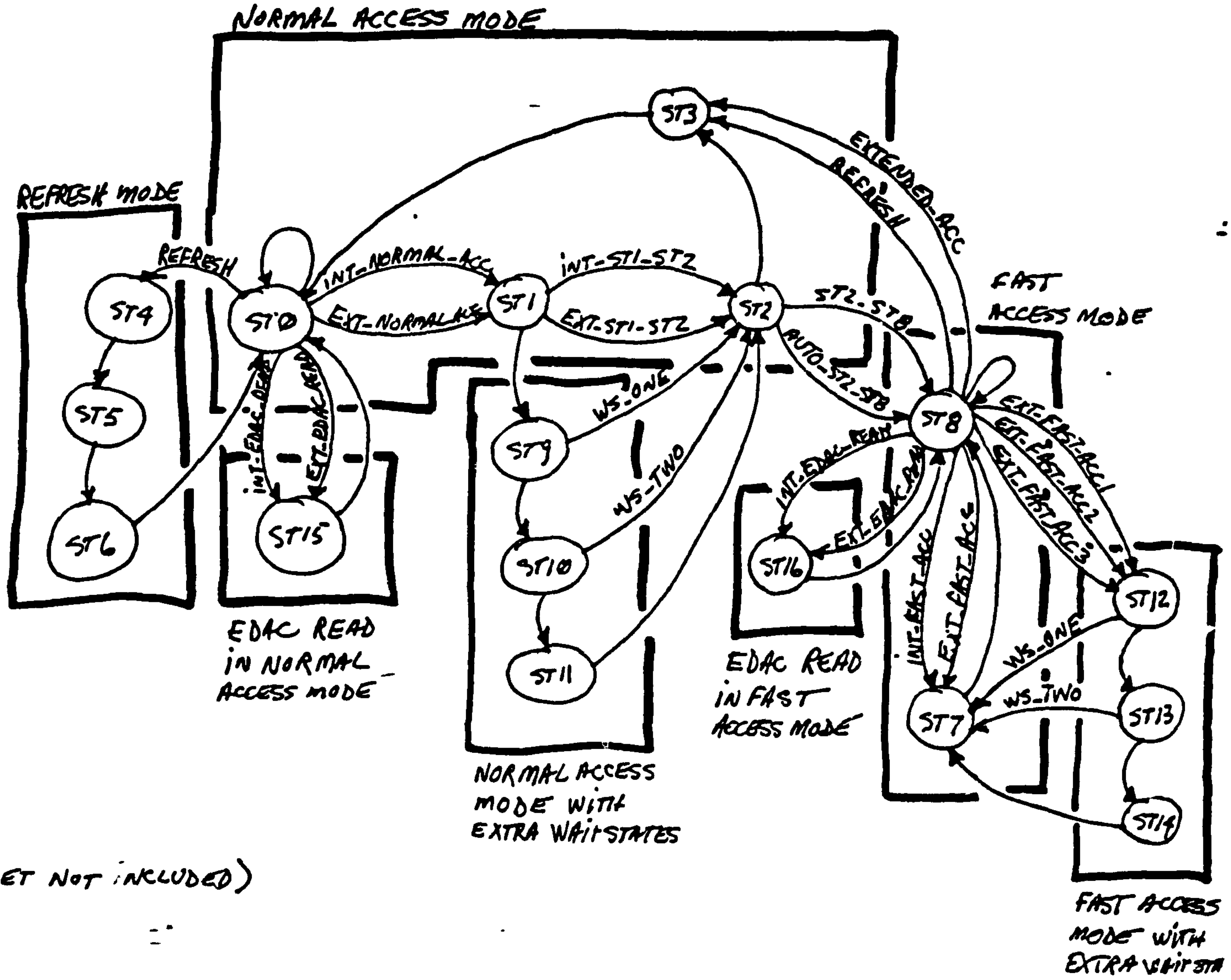
```
ST7                1      1      0      0      0      0      1      1
ST8                1      1      0     Xø      1      0      0      1
ST9                1      1      0      0      1      0      0      1
ST10               1      1      0      0      1      0      0      1
ST11               1      1      0      0      1      0      0      1
T12                1      1      0      0      1      0      0      1
                   1      1      0      0      1      0      0      1
S                  1      1      0      0      1      0      0      1
ST15               1      0      1      1      0      0      0      0
ST16               1      1      0    SXø      0      0      0      0

T0.OUTF   := /MSEL
T1.OUTF   := /RASI * /MSEL
T2.OUTF   := /READY * /RASI * /CASI
T3.OUTF   := /MSEL
T4.OUTF   := /MC1 * RFC * /RASI * /MSEL
T5.OUTF   := /MC1 * RFC * /RASI * /MSEL
T6.OUTF   := /MSEL
T7.OUTF   := /RASI * /CASI * /READY
T8.OUTF   := /RASI * /CAS;
T9.OUTF   := /RASI * /CASI
ST10.OUTF := /RASI * /CASI
ST11.OUTF := /RASI * /CASI
ST12.OUTF := /RASI * /CASI
ST13.OUTF := /RASI * /CASI
ST14.OUTF := /RASI * /CASI
ST15.OUTF := /MSEL * /READY * /OEDATA
ST16.OUTF := /RASI * /READY * /OEDATA  */CAS;

;*** STATE DESCRIPTIONS ***

      : Initial state prior to a refresh or a normal access cycle.
;-    : First state in the normal access cycle.
;ST2  : Second state in thé normal access cycle. Also last state in the
;       high-speed access (fast or static column decode access) when
;       the preceeding DRAM access was a normal or extended access.
;ST3  : Last state in the normal access cycle. Also garantees the required
;       RAS precharge time between two normal accesses, extended and normal
;       accesses, and between extended, normal accessed and refresh cycles.
;ST4  : First state in the refresh cycle.
;ST5  : Second state in the refresh cycle.
;ST6  : Last state in the refresh cycle.
;ST7  : Last state in the fast access cycle in the static column mode.
;ST8  : Initial state in the static column decode mode.
;ST9  : First extension state in the normal access cycle.
;ST10 : Second extension state in the normal access cycle.
;ST11 : Last extension state in the normal access cycle.
;ST12 : First extension state in the fast access cycle.
;ST13 : Second extension state in the fast access cycle.
;ST14 : Last extension state in the fast access cycle.
;ST15 : EDAC access state in the normal access mode.
;ST16 : EDAC access state in the fast access mode.

;*** STATE AND TRANSITION DEFINITIONS ***
POWER_UP := VCC -> STø               ;Go To State ø after power-up (or preset)
;* NORMAL ACCESS MODE *

0   := REFRESH         -> ST4  ;If refresh is requested then go to ST4
     + INT_NORMAL_ACC  -> ST1  ;Else if the local slow memory is accessed
     + EXT_NORMAL_ACC  -> ST1  ;Then go to ST1
     + INT_EDAC_READ   -> ST15 ;If there is an error and the processor reads
     + EXT_EDAC_READ   -> ST15 ;the local slow memory then go to ST15 (EDAC)
                               ;Else go to ST0.
```

```
T1   := INT_ST1_ST2         -> ST2  ;If it is a DRAM access with no extra state
      + EXT_ST1_ST2         -> ST2  ;Then go to ST2
      + /RESET              -> ST0  ;If there is a system reset then go to ST0
                           +-> ST9  ;Else go to ST9 for a DRAM access with extra
                                    ;states.
T2   := ST2_ST8             -> ST8  ;If static column decode is selected
      + AUTO_ST2_ST8        -> ST8  ;If the automatic mode is selected then ST8
      + /RESET              -> ST0  ;If there is a system reset then go to ST0
                           +-> ST3  ;Else completes the normal access cycle.

T3   := VCC                 -> ST0  ;Go to ST0.

 * REFRESH MODE *

ST4  := RESET               -> ST5  ;If no system reset then go to ST5
                                    ;Else go to ST0.
ST5  := RESET               -> ST6  ;If no system reset then go to ST6
                                    ;Else go to ST0.
ST6  := VCC                 -> ST0  ;Go to ST0 (this garantees precharge time)

;* FAST ACCESS MODE *

ST7  := RESET               -> ST8  ;If no system reset then go to ST8
                                    ;Else go to ST0.
ST8  := REFRESH             -> ST3  ;If refresh request then ST3 for precharge
      + EXTENDED_ACC        -> ST3  ;If not the same row then ST3 for precharge
      + INT_FAST_ACC        -> ST7  ;If DRAM access with no extra states
      + EXT_FAST_ACC        -> ST7  ;Then go to ST7
      + INT_EDAC_READ       -> ST16 ;If an error occured and the processor reads
      + EXT_EDAC_READ       -> ST16 ;at a DRAM address then go to ST16
      + /RESET              -> ST0  ;If there is a system reset then go to ST0
      + EXT_FAST_ACC1       -> ST12 ;If DRAM access with extra states then ST12
      + EXT_FAST_ACC2       -> ST12
      + EXT_FAST_ACC3       -> ST12
                           +-> ST8  ;Else there is no DRAM access then ST8.

;* NORMAL ACCESS MODE WITH EXTRA WAIT STATES *

ST9  := WS_ONE              -> ST2  ;If one extra state is selected then ST2
      + /RESET              -> ST0  ;If there is a system reset then go to ST0
                           +-> ST10 ;Else 2 or 3 extra states are performed.

ST10 := WS_TWO              -> ST2  ;If two extra states are selected then ST2
      + /RESET              -> ST0  ;If there is a system reset then go to ST0
                           +-> ST11 ;Else one more extra wait state is performed.

ST11 := RESET               -> ST2  ;If no system reset then go to ST2
                                    ;Else go to ST0.

;* FAST ACCESS MODE WITH EXTRA WAIT STATES *

ST12 := WS_ONE              -> ST7  ;If one extra state is selected then ST7
      + /RESET              -> ST0  ;If there is a system reset then go to ST0
                           +-> ST13 ;Else 2 or 3 extra states are performed.

ST13 := WS_TWO              -> ST7  ;If two extra states are selected then ST7
      + /RESET              -> ST0  ;If there is a system reset then go to ST0
                           +-> ST14 ;Else one more extra wait state is performed.

ST14 := RESET               -> ST7  ;If no system reset then go to ST7
                                    ;Else go to ST0.

;* EDAC READ IN NORMAL ACCESS MODE *
```

```
ST15 := VCC             -> ST0  ;Go to ST0.

;* EDAC READ IN FAST ACCESS MODE *

ST16 := RESET           -> ST8  ;If no system reset then go to ST8
                                ;Else go to ST0.

;*** BRANCH CONDITIONS ***

CONDITIONS

;* REFRESH *

REFRESH        = /REFREQ * RESET
;Refresh cycle is requested.

;* SLOW MEMORY ACCESSES *

INT_NORMAL_ACC = LOCAL_MEMORY_ACCESS * ERR * /SELECT ~~* REFREQ~~
;Normal access cycle to the 4 Mwords DRAM block.

EXT_NORMAL_ACC = LOCAL_MEMORY_ACCESS * ERR * SELECT * MEM ~~* REFREQ~~
;Normal access cycle to the memory extension.

INT_FAST_ACC   = LOCAL_MEMORY_ACCESS * ERR * /HSA * /SELECT ~~* REFREQ~~
;Fast access cycle to the 4 Mwords DRAM block.

EXT_FAST_ACC   = LOCAL_MEMORY_ACCESS * ERR * /HSA * SELECT * MEM * /WS0
                 * /WS1 ~~* REFREQ~~
;Fast access cycle to the memory extension.

EXT_FAST_ACC1  = LOCAL_MEMORY_ACCESS * ERR * /HSA * SELECT * MEM * WS0
                 * /WS1 ~~* REFREQ~~
;Fast access cycle to the memory extension with one extra state.

EXT_FAST_ACC2  = LOCAL_MEMORY_ACCESS * ERR * /HSA * SELECT * MEM * /WS0
                 * WS1 ~~* REFREQ~~
;Fast access cycle to the memory extension with two extra states.

EXT_FAST_ACC3  = LOCAL_MEMORY_ACCESS * ERR * /HSA * SELECT * MEM * WS0
                 * WS1 ~~* REFREQ~~
;Fast access cycle to the memory extension with three extra states.

EXTENDED_ACC   = LOCAL_MEMORY_ACCESS * ERR * HSA ~~* REFREQ~~
;Extended access cycle when leaving the static column decode mode.

;* EDAC READ CYCLES *

INT_EDAC_READ  = LOCAL_MEMORY_ACCESS * /ERR * /SELECT * RW ~~* REFREQ~~
;EDAC read cycle to the 4 Mwords DRAM block range addresses.

EXT_EDAC_READ  = LOCAL_MEMORY_ACCESS * /ERR * SELECT * MEM * RW ~~* REFREQ~~
;EDAC read cycle to the memory extension range addresses (if not memory ->
;not valid).

;* EXTENDED ACCESS CYCLES FOR NORMAL AND FAST ACCESSES *

WS_ONE         = WS0 * /WS1 * RESET ;+ one H clock cycle
WS_TWO         = /WS0 * WS1 * RESET ;+ two H clock cycles

;* INTER-STATES CONDITIONS *
```

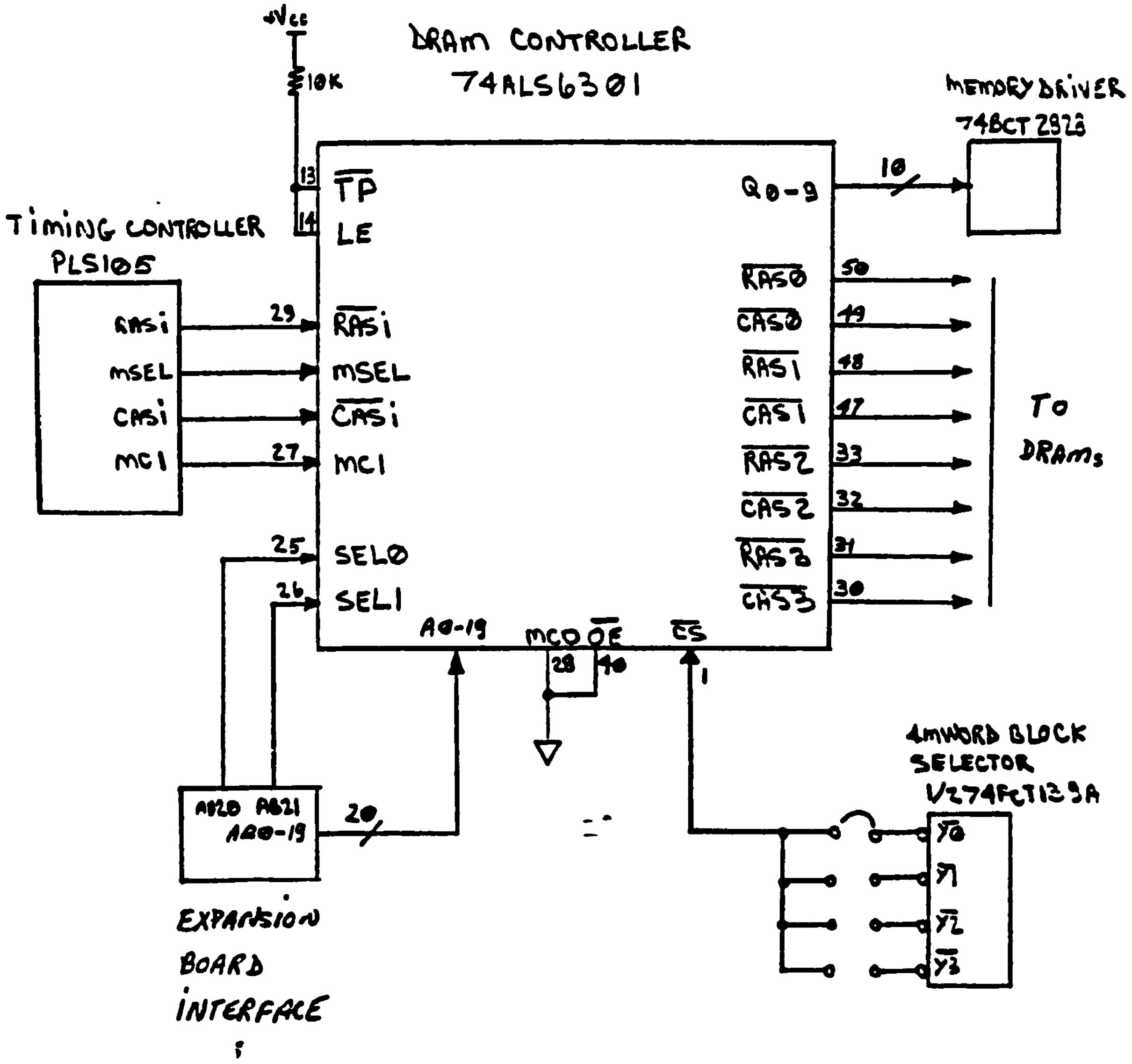
```
T_ST1_ST2      = /SELECT * RESET
T_ST1_ST2      = SELECT * /WS0 * /WS1 * RESET ;ST1 to ST2 in extension access
2_ST8          = /AUTO * SCD * RESET          ;ST2 to ST8 in SCD mode
TO_ST2_ST8     = AUTO * /HSA * SELECT * RESET ;ST2 to ST8 in automatic mode
                                                          and fast access

IATIONS *

GLOBAL.SETF = /RESET  ;Programmable preset function (all registers = 1)
```

TIMING CONTROLLER
STATE DIAGRAM

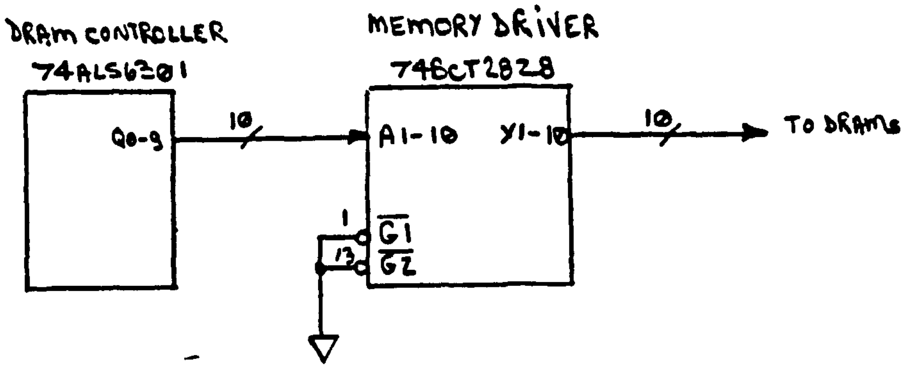


(RESET NOT INCLUDED)

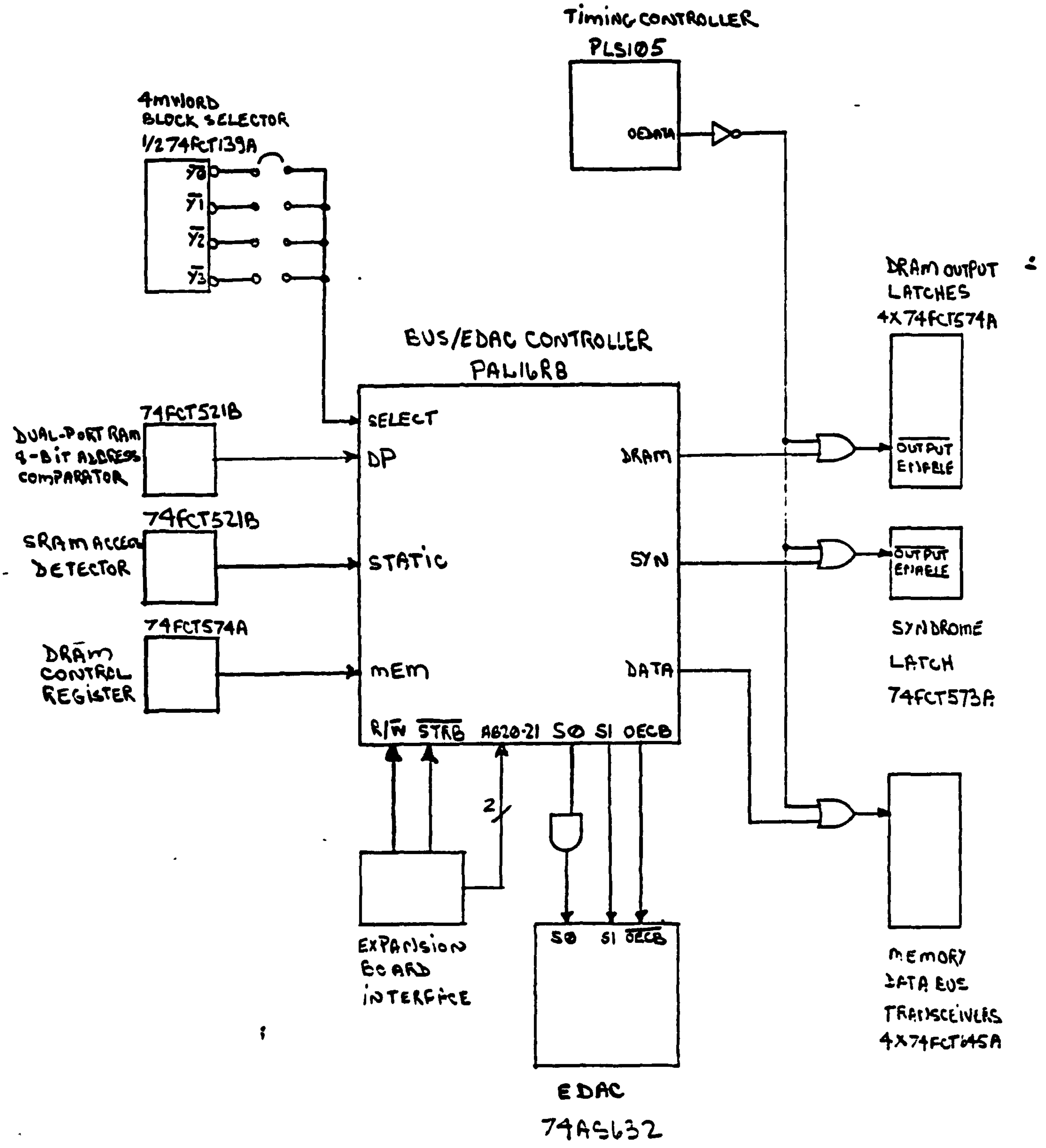
DRAM CONTROLLER



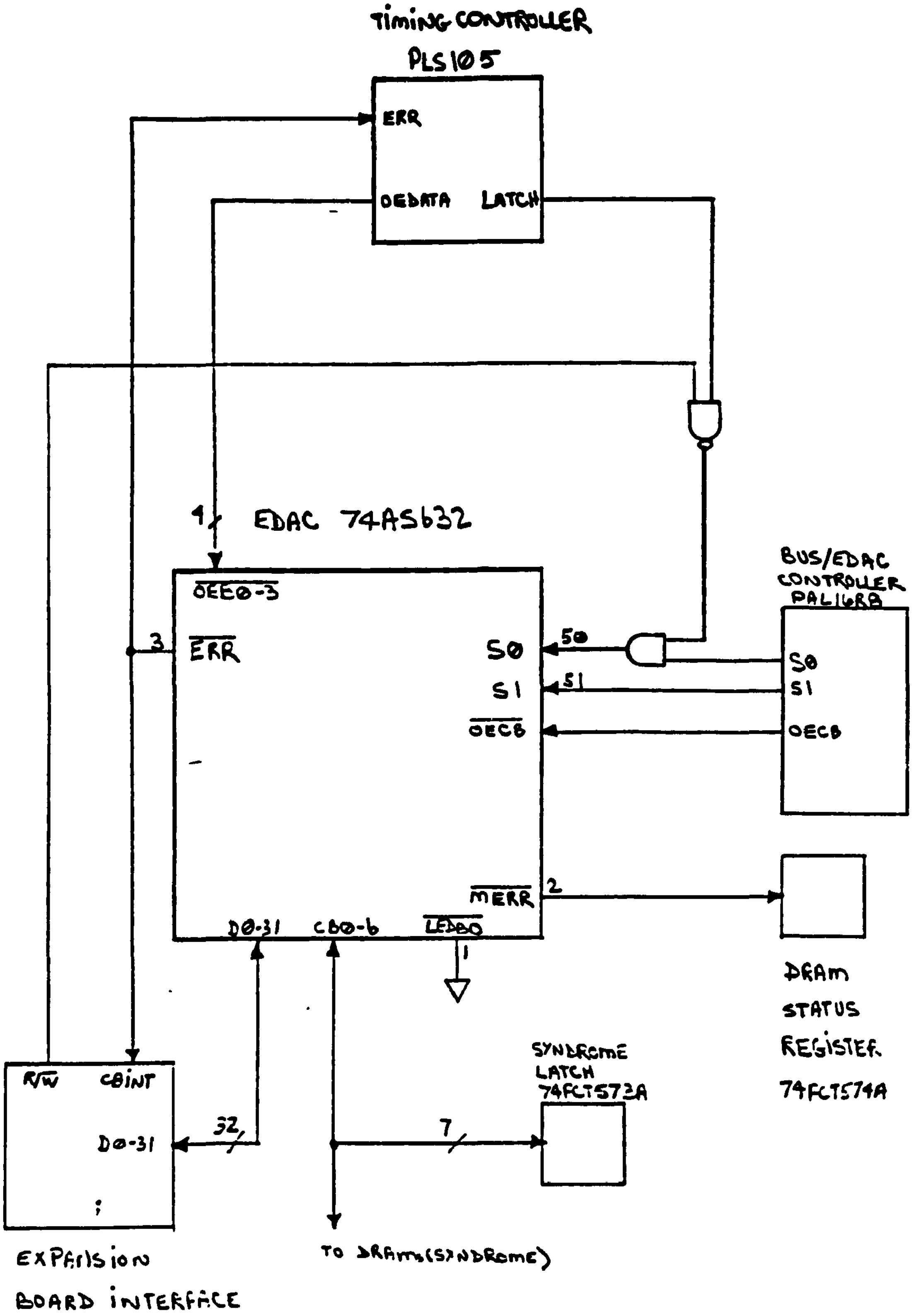
MEMORY DRIVER



BUS/EDAC CONTROLLER

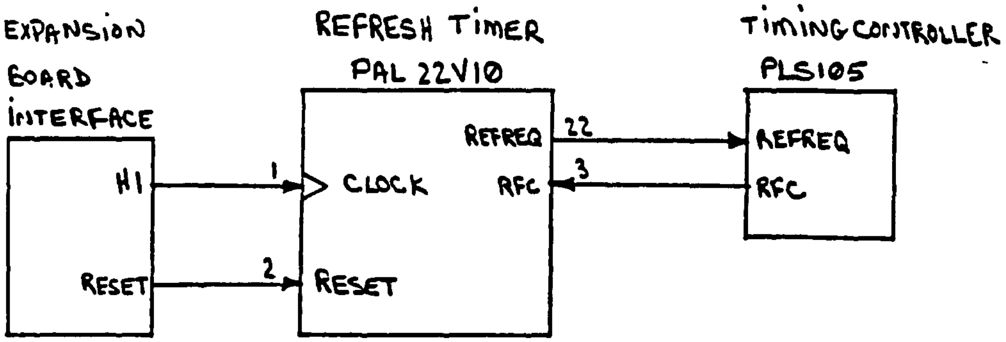


EDAC



TIMING CONTROLLER
PLS105
ERR
OEDATA
LATCH
4
EDAC 74AS632
$\overline{OEE0-3}$
3
$\overline{ERR}$
S0
S1
$\overline{OECB}$
$\overline{MERR}$
D0-31
CB0-6
$\overline{LEDBO}$
50
51
1
2
BUS/EDAC
CONTROLLER
PAL16R8
S0
S1
OECB
DRAM
STATUS
REGISTER
74FCT574A
SYNDROME
LATCH
74FCT573A
7
R/W
CBINT
D0-31
32
EXPANSION
BOARD INTERFACE
TO DRAMS(SYNDROME)

REFRESH TIMER



```
ℂLE     Refresh Timer

ESCRIPTION
ıe refresh timer is used to signal to the timing controller on the
ıS320C30 DSP board that it should execute a refresh cycle as required
, the dynamic memory. This device sends a REFREQ signal every 160
 ock cycles with a frequency of 16.5 MHz. This makes it compatible
 1Mx1 DRAMs (512 rows with every row being refreshed once every
 ms or less. The division rate of 160 provides extra time for timing
rbitration.

ℂTERN    REFRESH.PDS
/ISION   1
ℂHOR     S. Martel
ıPANY    McGill University
ℂE       7 August 1988

IP       REFRESH_TIMER  PAL22V10

INS
l      2      3     4     5     6     7     8     9     10      11    12
CLOCK  RESET  RFC   NC    NC    NC    NC    NC    NC    NC      NC    GND

l3     14     15    16    17    18    19    20    21    22      23    24
NC     Q0     Q1    Q2    Q3    Q4    Q5    Q6    Q7    REFREQ  NC    VCC
OBAL

LOCK: H1 at 16.5 MHz from the processor
ESET: System reset (active low)
FC  : Refresh complete signal from the timing controller (active high)
..  : Counter states (not connected)
EFREQ: Refresh request (output active low)

     CNT_160 ' /Q0 * /Q1 * /Q2 * /Q3 * /Q4 * Q5 * /Q6 * Q7 '
RING SCLR    ' /RESET + CNT_160 '

UATIONS

EFREQ := CNT_160 * RESET
       + /REFREQ * /RFC * RESET

     := /Q0  * /SCLR
     := (Q1  :+: Q0) * /SCLR
     := (Q2  :+: (Q1 * Q0)) * /SCLR
     := (Q3  :+: (Q2 * Q1 * Q0)) * /SCLR
     := (Q4  :+: (Q3 * Q2 * Q1 * Q0)) * /SCLR
     := (Q5  :+: (Q4 * Q3 * Q2 * Q1 * Q0)) * /SCLR
     := (Q6  :+: (Q5 * Q4 * Q3 * Q2 * Q1 * Q0)) * /SCLR
     := (Q7  :+: (Q6 * Q5 * Q4 * Q3 * Q2 * Q1 * Q0)) * /SCLR

ri-state outputs enable (optional)
FREQ.TRST = VCC
.TRST = VCC
.TRST = VCC
.TRST = VCC
.TRST = VCC
.TRST = VCC
.TRST = VCC
.TRST = VCC
  ST = VCC

OBAL.RSTF = /RESET ;Programmable reset function (all register outputs = 0)

MULATION

ACE_ON RESET CLOCK RFC REFREQ Q0 Q1 Q2 Q3 Q4 Q5 Q6 Q7
```

F14

```
SETF /CLOCK
CLOCKF CLOCK
SETF /RESET /RFC
CLOCKF CLOCK
CLOCKF CLOCK
SETF RESET
FOR C:=1 TO 16 DO  ;start counting
  BEGIN
  CLOCKF CLOCK
  END
CHECK REFREQ
PRLDF /Q0 Q1 Q2 Q3 Q4 /Q5 /Q6 Q7  ;preload registers
FOR C:=1 TO 5 DO  ;count in order to reah 160
  BEGIN
  CLOCKF CLOCK
  END
CHECK /REFREQ
SETF RFC
CLOCKF CLOCK
CHECK REFREQ
FOR C:=1 TO 5 DO
  BEGIN
  CLOCKF CLOCK
  END
CHECK REFREQ
SETF /RFC
CLOCKF CLOCK
CHECK REFREQ

TRACE_OFF
```

```
ALASM XPLOT, V2.23 - MARKET RELEASE (2-1-88)
C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988

itle      : Refresh Timer                  Author  : S. Martel
attern    : REFRESH.PDS                         Company : McGill University
evision : 1                                    Date    : 7 August 1988

V10
EFRESH_TIMER

                 11 1111 1111 2222 2222 2233 3333 3333 4444
   0123 4567 8901 2345 6789 0123 4567 8901 2345 6789 0123

 0 ---- -X-- ---- ---- ---- ---- ---- ---- ---- ---- ----

 1 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 2 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 3 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 4 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 5 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 6 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 7 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 8 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 9 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

10 ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
11 ---- X--- ---X --X- ---X --X- --X- --X- --X- --X- ----
12 ---- X--X -X-- ---- ---- ---- ---- ---- ---- ---- ----
13 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
14 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
15 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
16 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
17 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
    XXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
19 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
20 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

21 ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
22 ---- X--- ---X ---- ---- ---- ---- ---- --X- ---- ----
23 ---- X--- ---X ---- ---- --X- ---- ---- ---- ---- ----
24 ---- X--- ---X ---- --X- ---- ---- ---- ---- ---- ----
25 ---- X--- ---X ---- ---- ---- ---- --X- ---- ---- ----
26 ---- X--- ---X ---- ---- ---- --X- ---- ---- ---- ----
27 ---- X--- ---X --X- ---- ---- ---- ---- ---- ---- ----
28 ---- ---- ---X --X- ---X --X- --X- --X- --X- --X- ----
29 ---- X--- --X- ---X ---X ---X ---X ---X ---X ---X ----
30 ---- X--- ---X ---- ---- ---- ---- ---- ---- --X- ----
31 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
32 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
33 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

34 ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
35 ---- X--- ---- ---X ---- ---- --X- ---- ---- ---- ----
36 ---- X--- ---- ---X ---- --X- ---- ---- ---- ---- ----
37 ---- X--- ---- ---X ---- ---- ---- ---- --X- ---- ----
38 ---- X--- ---- ---X ---- ---- ---- --X- ---- ---- ----
39 ---- X--- ---- ---X --X- ---- ---- ---- ---- ---- ----
40 ---- X--- ---- --X- ---X ---X ---X ---X ---X ---X ----
41 ---- ---- ---X --X- ---X --X- --X- --X- --X- --X- ----
   ---- X--- ---- ---X ---- ---- ---- ---- ---- --X- ----
   XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
44 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
45 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
46 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
47 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
48 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
```

F16

```
 49 ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
 50 ---- X--- ---- ---- ---X ---- --X- ---- ---- ---- ----
 51 ---- X--- ---- ---- ---X ---- ---- ---- --X- ---- ----
 52 ---- X--- ---- ---- ---X ---- ---- --X- ---- ---- ----
 53 ---- X--- ---- ---- ---X --X- ---- ---- ---- ---- ----
 54 ---- X--- ---- ---- --X- ---X ---X ---X ---X ---X ----
 55 ---- ---- ---X --X- ---X --X- --X- --X- --X- --X- ----
 56 ---- X--- ---- ---- ---X ---- ---- ---- ---- --X- ----
 57 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 58 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 59 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 60 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 61 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 62 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 63 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 64 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 65 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 66 ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
 67 ---- X--- ---- ---- ---- ---X ---- ---- ---- --X- ----
 68 ---- X--- ---- ---- ---- ---X ---- ---- --X- ---- ----
 69 ---- X--- ---- ---- ---- ---X --X- ---- ---- ---- ----
 70 ---- X--- ---- ---- ---- --X- ---X ---X ---X ---X ----
 71 ---- ---- ---X --X- ---X --X- --X- --X- --X- --X- ----
 72 ---- X--- ---- ---- ---- ---X ---- --X- ---- ---- ----
 73 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 74 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 75 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 76 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 77 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 78 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 79 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 80 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 81 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 82 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 83 ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
 84 ---- X--- ---- ---- ---- ---- ---X ---- --X- ---- ----
 85 ---- X--- ---- ---- ---- ---- ---X --X- ---- ---- ----
 86 ---- X--- ---- ---- ---- ---- --X- ---X ---X ---X ----
 87 ---- ---- ---X --X- ---X --X- --X- --X- --X- --X- ----
 88 ---- X--- ---- ---- ---- ---- ---X ---- ---- --X- ----
 89 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 90 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 91 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 92 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 93 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 94 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 95 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 96 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 97 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 98 ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
 99 ---- X--- ---- ---- ---- ---- ---- ---X --X- ---- ----
100 ---- X--- ---- ---- ---- ---- ---- --X- ---X ---X ----
101 ---- ---- ---X --X- ---X --X- --X- --X- --X- --X- ----
102 ---- X--- ---- ---- ---- ---- ---- ---X ---- --X- ----
103 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
104 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
105 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
106 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
107 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
108 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
109 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
110 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
```

```
111 ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
112 ---- X--- ---- ---- ---- ---- ---- ---- --X- ---X ----
113 ---- ---- ---X --X- ---X --X- --X- --X- --X- --X- ----
114 ---- X--- ---- ---- ---- ---- ---- ---- ---X --X- ----
115 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
116 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
117 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
118 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
119 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
120 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
121 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

122 ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
123 ---- ---- ---X --X- ---X --X- --X- --X- --X- --X- ----
124 ---- X--- ---- ---- ---- ---- ---- ---- ---- --X- ----
125 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
126 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
127 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
128 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
129 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
130 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

131 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX


      OUTPUT PINS: 1111112222
                   4567890123
    POLARITY FUSE: --------XX

      OUTPUT PINS: 1111112222
                   4567890123
       FLUSH FUSE: XXXXXXXXXX

TOTAL FUSES BLOWN:  2267
```

```
PALASM XPLOT, V2.23 - MARKET RELEASE (2-1-88)
(C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988

Title    : Refresh Timer                     Author  : S. Martel
Pattern  : REFRESH.PDS                       Company : McGill University
Revision : 1                                 Date    : 7 August 1988

PAL22V10
REFRESH_TIMER*
QV512*
QP24*
QF5828*
G0*F0*
L0000 11111011111111111111111111111111111111111111*
L0440 11111111111111111111111111111111111111111111*
L0484 11110111111011011110110111011101110111011111*
L0528 11110110101111111111111111111111111111111111*
L0924 11111111111111111111111111111111111111111111*
L0968 11110111111011111111111111111111111110111111111*
L1012 11110111111011111111110111111111111111111111*
L1056 11110111111011111101111111111111111111111111*
L1100 11110111111011111111111111111101111111111111*
L1144 11110111111011111111111111110111111111111111*
L1188 11110111111011011111111111111111111111111111*
L1232 11111111111011011110110111011101110111011111*
L1276 11110111110111101110111011101110111011101111*
L1320 11110111111011111111111111111111111111011111*
L1496 11111111111111111111111111111111111111111111*
L1540 11110111111111101111111111011111111111111111*
L1584 11110111111111101111110111111111111111111111*
L1628 11110111111111101111111111111111110111111111*
L1672 11110111111111101111111111111101111111111111*
L1716 11110111111111101101111111111111111111111111*
L1760 11110111111111011110111011101110111011101111*
L1804 11111111111011011110110111011101110111011111*
L1848 11110111111111101111111111111111111111011111*
L2156 11111111111111111111111111111111111111111111*
L2200 11110111111111111110111111011111111111111111*
L2244 11110111111111111110111111111111110111111111*
L2288 11110111111111111110111111111101111111111111*
L2332 11110111111111111110110111111111111111111111*
L2376 11110111111111111101111011101110111011101111*
L2420 11111111111011011110110111011101110111011111*
L2464 11110111111111111110111111111111111111011111*
L2904 11111111111111111111111111111111111111111111*
L2948 11110111111111111111111011111111111111011111*
L2992 11110111111111111111111011111111110111111111*
L3036 11110111111111111111111011011111111111111111*
L3080 11110111111111111111110111101110111011101111*
L3124 11111111111011011110110111011101110111011111*
L3168 11110111111111111111111011111101111111111111*
L3652 11111111111111111111111111111111111111111111*
L3696 11110111111111111111111111110111111011111111*
L3740 11110111111111111111111111110110111111111111*
L3784 11110111111111111111111111011110111011101111*
L3828 11111111111011011110110111011101110111011111*
L3872 11110111111111111111111111110111111111011111*
L4312 11111111111111111111111111111111111111111111*
L4356 11110111111111111111111111111110110111111111*
L4400 11110111111111111111111111111101111011101111*
L4444 11111111111011011110110111011101110111011111*
L4488 11110111111111111111111111111110111111011111*
L4884 11111111111111111111111111111111111111111111*
L4928 11110111111111111111111111111111110111101111*
L4972 11111111111011011110110111011101110111011111*
```

```
6016 11110111111111111111111111111111111011011111*
6368 11111111111111111111111111111111111111111111*
6412 11111111111011011110110111011101110111011111*
6456 11110111111111111111111111111111111111011111*
6808 0000101010*
6818 1010101010*
     CXXXXXXXXXXNXXXXXXXXXXXN*
     C00XXXXXXXXNXLLLLLLLLHXN*
0003 C00XXXXXXXXNXLLLLLLLLHXN*
0004 C10XXXXXXXXNXHLLLLLLLHXN*
0005 C10XXXXXXXXNXLHLLLLLLHXN*
0006 C10XXXXXXXXNXHHLLLLLLHXN*
0007 C10XXXXXXXXNXLLHLLLLLHXN*
0008 C10XXXXXXXXNXHLHLLLLLHXN*
0009 C10XXXXXXXXNXLHHLLLLLHXN*
0010 C10XXXXXXXXNXHHHLLLLLHXN*
0011 C10XXXXXXXXNXLLLHLLLLHXN*
0012 C10XXXXXXXXNXHLLHLLLLHXN*
0013 C10XXXXXXXXNXLHLHLLLLHXN*
0014 C10XXXXXXXXNXHHLHLLLLHXN*
0015 C10XXXXXXXXNXLLHHLLLLHXN*
0016 C10XXXXXXXXNXHLHHLLLLHXN*
0017 C10XXXXXXXXNXLHHHLLLLHXN*
0018 C10XXXXXXXXNXHHHHLLLLHXN*
0019 C10XXXXXXXXNXLLLLHLLLHXN*
0020 P10XXXXXXXXNX100001101XN*
0021 C10XXXXXXXXNXHHHHHLLHHXN*
0022 C10XXXXXXXXNXLLLLLHLHHXN*
0023 C10XXXXXXXXNXHHHHHHHHLXN*
0024 C10XXXXXXXXNXLLLLLLLLLXN*
0025 C10XXXXXXXXNXHLLLLLLLLXN*
0026 C11XXXXXXXXNXLHLLLLLLHXN*
0027 C11XXXXXXXXNXHHLLLLLLHXN*
     C11XXXXXXXXNXLLHLLLLLHXN*
0029 C11XXXXXXXXNXHLHLLLLLHXN*
0030 C11XXXXXXXXNXLHHLLLLLHXN*
0031 C11XXXXXXXXNXHHHLLLLLHXN*
0032 C10XXXXXXXXNXLLLHLLLLHXN*
C0D27*
68AA
```

F20

```
LASM SIMULATION, V2.23 - MARKET RELEASE ([illegible]
) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988
LASM SIMULATION SELECTIVE TRACE LISTING

itle     : Refresh Timer            Author    : S. Martel
tern     : REFRESH.PDS              Company   : McGill University
    on : 1                          Date      : 7 August 1988

AL22V10
EFRESH_TIMER
age :  1
        gcg c cg   c  c  c  c   c  c  c   c  c  c
RESET   XXLLLLLHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
CLOCK   HLLHLHLLHH LHHLHHLHHL HHLHHLHHLH HLHHLHHLHH
RFC     XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
REFREQ  XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
Q0      XXLLLLLLLH HHLLLHHHLL LHHHLLLHHH LLLHHHLLLH
Q1      XXLLLLLLLL LLHHHHHHLL LLLLHHHHHH LLLLLLHHHH
Q2      XXLLLLLLLL LLLLLLLLHH HHHHHHHHHH LLLLLLLLLL
Q3      XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL HHHHHHHHHH
Q4      XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
Q5      XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
Q6      XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
Q7      XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
```

```
AL22V10
EFRESH_TIMER
age :  2
         c  c  c  c   c  cpg c   c  c  c   cg  c  c
RESET    HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
CLOCK    LHHLHHLHHL HHLHHLLHHL HHLHHLHHLH HLLHHLHHLH
         LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLHHHHHHHH
      EQ HHHHHHHHHH HHHHHHHHHH HHHHLLLLLL LLLLHHHHHH
Q0       HHLLLHHHLL LHHHLLLLHH HLLLHHHLLL HHHHLLLHHH
Q1       HHLLLLLLHH HHHHLLHHHH HLLLHHHLLL LLLLHHHHHH
Q2       LLHHHHHHHH HHHHLLHHHH HLLLHHHLLL LLLLLLLLLL
Q3       HHHHHHHHHH HHHHLLHHHH HLLLHHHLLL LLLLLLLLLL
Q4       LLLLLLLLLL LLLLHHHHHH HLLLHHHLLL LLLLLLLLLL
Q5       LLLLLLLLLL LLLLLLLLLL LHHHHHHLLL LLLLLLLLLL
Q6       LLLLLLLLLL LLLLLLLLLL LLLLHHHLLL LLLLLLLLLL
Q7       LLLLLLLLLL LLLLLLHHHH HHHHHHHLLL LLLLLLLLLL
```

```
PAL22V10
REFRESH_TIMER
Page :  3
         c  c  c   cg  c
 RESET  HHHHHHHHHH HHHHH
 CLOCK  HLHHLHHLHH LLHHL
 [illegible] HHHHHHHHHH HLLLL
 [illegible]REQ HHHHHHHHHH HHHHH
 Q0     LLLHHHLLLH HHHLL
 Q1     LLLLLLHHHH HHHLL
 Q2     HHHHHHHHHH HHHLL
 Q3     LLLLLLLLLL LLLHH
 Q4     LLLLLLLLLL LLLLL
 Q5     LLLLLLLLLL LLLLL
 Q6     LLLLLLLLLL LLLLL
 Q7     LLLLLLLLLL LLLLL
```

```
PALASM SIMULATION, V2.23 - MARKET RELEASE (2-1-88)
(C) - COPYRIGHT MONOLITHIC MEMORIES INC, 1988
PALASM SIMULATION HISTORY LISTING

Title     : Refresh Timer           Author    : S. Martel
Pattern   : REFRESH.PDS             Company   : McGill University
Revision  : 1                       Date      : 7 August 1988

PAL22V10
REFRESH_TIMER
Page :  1
        gcg c cg   c  c  c  c   c  c  c   c  c  c
 CLOCK  HLLHLHLLHH LHHLHHLHHL HHLHHLHHLH HLHHLHHLHH
 RESET  XXLLLLLHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 RFC    XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 GND    LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 Q0     XXLLLLLLLH HHLLLHHHLL LHHHLLLHHH LLLHHHLLLH
 Q1     XXLLLLLLLL LLHHHHHHLL LLLLHHHHHH LLLLLLHHHH
 Q2     XXLLLLLLLL LLLLLLLLHH HHHHHHHHHH LLLLLLLLLL
 Q3     XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL HHHHHHHHHH
 Q4     XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 Q5     XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 Q6     XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 Q7     XXLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 REFREQ XXHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 VCC    HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 GLOBAL XXLLLLLXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX
```

```
PAL22V10
REFRESH_TIMER
Page : 2
        c  c  c  c   c  cpg c   c  c  c   cg  c  c
 CLOCK  LHHLHHLHHL HHLHHLLHHL HHLHHLHHLH HLLHHLHHLH
 RESET  HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
        LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLHHHHHHHH
        LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL LLLLLLLLLL
 Q0     HHLLLHHHLL LHHHLLLLHH HLLLHHHLLL HHHHLLLHHH
 Q1     HHLLLLLLHH HHHHLLHHHH HLLLHHHLLL LLLLHHHHHH
 Q2     LLHHHHHHHH HHHHLLHHHH HLLLHHHLLL LLLLLLLLLL
 Q3     HHHHHHHHHH HHHHLLHHHH HLLLHHHLLL LLLLLLLLLL
 Q4     LLLLLLLLLL LLLLHHHHHH HLLLHHHLLL LLLLLLLLLL
 Q5     LLLLLLLLLL LLLLLLLLLL LHHHHHHLLL LLLLLLLLLL
 Q6     LLLLLLLLLL LLLLLLLLLL LLLLHHHLLL LLLLLLLLLL
 Q7     LLLLLLLLLL LLLLLLHHHH HHHHHHHLLL LLLLLLLLLL
 REFREQ HHHHHHHHHH HHHHHHHHHH HHHHLLLLLL LLLLHHHHHH
 VCC    HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH HHHHHHHHHH
 GLOBAL XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX XXXXXXXXXX
```

```
AL22V10
EFRESH_TIMER
age :  3
          c  c   c    cg  c
CLOCK    HLHHLHHLHH LLHHL
RESET    HHHHHHHHHH HHHHH
         HHHHHHHHHH HLLLL
         LLLLLLLLLL LLLLL
Qv       LLLHHHLLLH HHHLL
Q1       LLLLLLHHHH HHHLL
Q2       HHHHHHHHHH HHHLL
Q3       LLLLLLLLLL LLLHH
Q4       LLLLLLLLLL LLLLL
Q5       LLLLLLLLLL LLLLL
Q6       LLLLLLLLLL LLLLL
Q7       LLLLLLLLLL LLLLL
REFREQ   HHHHHHHHHH HHHHH
VCC      HHHHHHHHHH HHHHH
GLOBAL   XXXXXXXXXX XXXXX
```

APPENDIX G - UNIT LIST

Unit number	Type	Sheet
U1	TMS320C30	A1
U2-U5	74FCT138A	A2
U6	74FCT521B	A3
U7-U10	74FCT645A	A3
U11-U14	74FCT244A	A3
U15A	74F00	A3
U15B	74F00	A6
U15C-U15D	(NOT USED)	
U16A	74FCT240A	A3
U16B	74FCT240A	A4
U16C	74FCT240A	A22
U16D	74FCT240A	B1
U16E	74FCT240A	D9
U16F-U16H	(NOT USED)	
U17-U20	74FCT574A	A4
U21	74FCT244A	A4
U22A	74FCT139A	A4
U22B	(NOT USED)	
U23	74FCT521B	A4
U24-U29	74FCT138A	A4
U30-U37	74F257	A5
U38	16R6	A6
U39A-U39B	74F32	A6
U39C-U39D	(NOT USED)	
U40A-U40B	74F08	A6
U40C	74F08	A22
U40D	(NOT USED)	
U41A	74F21	A6
U41B	(NOT USED)	
U42-U43	74F30	A6
U44-U45	16R8	A6
U46-U51	74FCT574A	A20

Unit number	**Type**	**Sheet**
U52-U53	74FCT273A	A20
U54	74FCT240	A20
U55-U57	74FCT273A	A21
U58	74FCT240	A21
U59	74FCT827B	A21
U60	74FCT244A	A21
U61-U68	74FCT645A	A21
U69-U71	74FCT244	A22
U72A	74LS74	A22
U72B	74LS74	A20
U73-U76	74FCT244A	B1
U77-U80	74FCT521B	B1
U81-U84	74FCT645A	B1
U85A	74ALS21A	B1
U85B	74ALS21A	B2
U86-U89	74FCT244A	B2
U90-U93	74FCT521B	B2
U94-U97	74FCT645A	B2
U98-U161	MT5C2564-25 (OR EQUIVALENT)	B3-B4
U162	IDT71321-35 OR CY7C136-35	C1
U163-U165	IDT71421-35 OR CY7C146-35	C2-C4
U166-U173	74FCT645A	C1-C4
U174-U176	74FCT244A	C5
U177	74FCT240A	C5
U178A-U178B	74ALS09	C5
U178C-U178D	(NOT USED)	
U179-U242	74FCT299	D1-D8
U243-U258	74LS592	D1-D8
U259-U266	74ALS113A	D1-D8
U267-U273	74FCT240	D1-D8
U274-U275	74FCT240A	D1-D8
U276-U279	74F08	D1-D8
U280-U287	74F32	D1-D8
U288-U292	74F11	D1-D8

Unit number	Type	Sheet
U293A	74F11	D8
U293B-U293C	(NOT USED)	
U294	74FCT521B	D9
U295-U301	74FCT138A	D9
U302-U333	74LS592	D10-D13
U334-U337	74FCT240	D10-D13
U338-U345	74ALS86	D10-D13
U346-U353	74FCT574A	D14
U354-U361	74ALS32	D15
U362-U381	AM26LS31C (OR EQUIVALENT)	D16-D18
U382-U389	AM26LS33AC (33C, 32C, OR EQUIVALENT)	D19
U390-U395	74FCT574A	D20
U396-U399	74ALS32	D20
U400-U403	74FCT244	E1
U404-U407	74FCT161	E1
U408-U409	74FCT244	E1
U410A	74FCT240	E1
U410B	74FCT240	E2
U410C	74FCT240	E3
U410D-U410H	74FCT240	E4
U411-U412	74FCT244	E2
U413-U416	74FCT574A	E2
U417A-U417B	74LS74	E2
U418A	74ALS08	E2
U418B-U418C	74ALS08	E3
U418D	(NOT USED)	
U419A-U419C	74F32	E2
U419D	(NOT USED)	
U420A	74F00	E2
U420B-U420C	74F00	E3
U420D	74F00	E4
U421A	74LS74	E2
U421B	(NOT USED)	
U422-U427	74FCT574A	E3

Unit number	Type	Sheet
U428A-U428B	74LS74	E3
U429	74FCT574A	E4
U430A-U430B	74F27	E4
U430C	(NOT USED)	
U431A-U431B	74F32	E4
U431C-U431D	(NOT USED)	
U432A-U432B	74F08	E4
U432C-U432D	(NOT USED)	
U433	74FCT244A	A20

APPENDIX H

EXTERNAL REGISTERS

Word length register 0 (write only)
initial address: 800010H
initial content: 00000000000000000000000000000000
XXXXXXXXXXXXXXXXXXXXXXXX00000000: word in channel 0 = 255 bits
XXXXXXXXXXXXXXXXXXXXXXXX11111111: word in channel 0 = 0 bit
00000000XXXXXXXXXXXXXXXXXXXXXXXX: word in channel 3 = 255 bits
11111111XXXXXXXXXXXXXXXXXXXXXXXX: word in channel 3 = 0 bit

Word length registers 1 to 3 (write only)
initial addresses: 800011H to 800013H
word length register 1: serial channels 4 to 7
word length register 2: serial channels 8 to 11
word length register 3: serial channels 12 to 15

Function in register (write only)
initial address: 800014H
initial content: XXXXXXXX
000000XX: general purpose status bits sent to the MicroVAX
XXXXXX0X: external interrupt to the MicroVAX
XXXXXXX0: clear FIFO channels 0 and 1 of the DRQ3B

Configuration register (write only)
initial address: 800015H
initial content: XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX0: disable external clock for channel 0 when RCO is low
XXXXXXXXXXXXXXX1: external clock always enable (normally non-continuous clk)
0XXXXXXXXXXXXXXX: disable external clock for channel 15 when RCO is low
1XXXXXXXXXXXXXXX: external clock always enable (normally non-continuous clk)

EXTERNAL REGISTERS

Serial control register (write only)
initial address: 800016H
initial content: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX00: hold data in serial channel 0
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX01: enable shift right in serial channel 0
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX10: enable shift left (normally not used)
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX11: parallel load (serial channel 0)
00XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX: hold data in serial channel 15
01XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX: enable shift right in serial channel 15
10XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX: enable shift left (normally not used)
11XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX: parallel load (serial channel 15)

Polarity register (write only)
initial address: 800017H
initial content: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX0: positive pulse on control line 0A
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX1: negative pulse on control line 0A
0XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX: positive pulse on control line 15B
1XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX: negative pulse on control line 15B

Pulse length register 0 (write only)
initial address: 800018H
initial content: 00000000000000000000000000000000
XXXXXXXXXXXXXXXXXXXXXXXX00000000: pulse length on control line 0A = 15.3 ms
XXXXXXXXXXXXXXXXXXXXXXXX11111110: pulse length on control line 0A = 60 ns
XXXXXXXXXXXXXXXXXXXXXXXX11111111: pulse length on control line 0A = 0 ns
00000000XXXXXXXXXXXXXXXXXXXXXXXX: pulse length on control line 1B = 15.3 ms
11111110XXXXXXXXXXXXXXXXXXXXXXXX: pulse length on control line 1B = 60 ns
11111111XXXXXXXXXXXXXXXXXXXXXXXX: pulse length on control line 1B = 0 ns

EXTERNAL REGISTERS

Pulse length registers 1 to 7 (write only)
initial addresses: 800019H to 80001FH
pulse length register 1: control lines 2 and 3
pulse length register 2: control lines 4 and 5
pulse length register 3: control lines 6 and 7
pulse length register 4: control lines 8 and 9
pulse length register 5: control lines 10 and 11
pulse length register 6: control lines 12 and 13
pulse length register 7: control lines 14 and 15

Start control pulses (write only)
initial address: 800020H
initial content: not applicable
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX0: start pulse on control line 0A
0XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX: start pulse on control line 15B

Mask register (write only)
initial address: 800021H
initial content: XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX1: mask interrupt from serial channel 0
1XXXXXXXXXXXXXXX: mask interrupt from serial channel 15

Multicast register (write only)
initial address: 800022H
initiai content: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX0: multicast write to dual-port module 0
0XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX: multicast write to dual-port module 30
* multicast register is only 31 uits

EXTERNAL REGISTERS

Wait-state register (write only)
initial address: 800023H
initial content: 01111
XXXX0: read dual-port module with 1 wait-state (no BUSY)
XXXX1: read dual-port module with 2 wait-states (no BUSY)
XXX0X: read dual-port module with 1 wait-state (with BUSY)
XXX1X: read dual-port module with 2 wait-states (with BUSY)
0X0XX: write to dual-port module(s) with 1 wait-state (no BUSY)
0X1XX: write to dual-port module(s) with 2 wait-states (no BUSY)
00XXX: write to dual-port module(s) with 1 wait-state (with BUSY)
01XXX: write to dual-port module(s) with 2 wait-states (with BUSY)
1XXXX: write to dual-port module(s) with 0 wait-state

Allocation register (write only)
initial address: 800024H
initial content: 00000000000000011111
XXXXXXXXXXXXXXX00000: SRAM block locations 000000H to 07FFFFH
XXXXXXXXXXXXXXX11111: SRAM block locations F80000H to FFFFFFH
XXXXXXX00000000XXXXX: dual-port module locations 000000H to 00FFFFH
XXXXXXX11111111XXXXX: dual-port module locations FF0000H to FFFFFFH
0000000XXXXXXXXXXXXX: I/O subsystem and external registers (8000000H-801FFFH)

Control register (write only)
initial address: 800025H
initial content: 000Y1000 (Y=0 for the communication processor, otherwise Y=1)
XXXXXXX0: disable interrupt controller (reset)
XXXXXX0X: serial channels 0 and 1 only have interrupt facilities (int. reg. not req.)
XXXXXX1X: all serial channels have interrupt facilities (interrupt reg. required)
XXXXX0XX: access the DPMs without considering any access conflicts
XXXXX1XX: allow the wait state generator to add a wait state during a conflict
XXXX1XXX: put the RS422 drivers in high impedance state
XXX0XXXX: put the connection bus in high impedance state
1XXXXXXX: allow transfers with the MicroVAX only

EXTERNAL REGISTERS

I/O subsytem initialization (write only)
initial address: 800026H
initial content: not applicable
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX: initialize I/O subsystem (reset)

Communication register (write only)
initial address: 800027H
initial content: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
00000000000000000000000000000000: send 00000000H to DRQ3B
11111111111111111111111111111111: send FFFFFFFFH to DRQ3B

Status register (read only)
initial address: 800010H
initial content: XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX0: data valid (in) or transmission complete (out) (channel 0)
0XXXXXXXXXXXXXXX: data valid (in) or transmission complete (out) (channel 15)

Interrupt register (read only)
initial address: 800011H
initial content: XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX0: interruption from serial channel 0
0XXXXXXXXXXXXXXX: interruption from serial channel 15

Pulse state register (read only)
initial address: 800012H
initial content: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX0: end of the pulse for control line 0A
0XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX: end of the pulse for control line 15B

EXTERNAL REGISTERS

Interface register (read only)
initial address: 800013H
initial content: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
00000000000000000000000000000000: receive 00000000H from DRQ3B
11111111111111111111111111111111: receive FFFFFFFFH from DRQ3B

Function out register (read only)
initial address: 800014H
initial content: XXXXXXX
XXXX000: general purpose status bits from the MicroVAX
XX00XXX: DRQ3B is initialized (reset)
X1XXXXX: data is waiting in the interface register
1XXXXXX: the communication register is ready to receive a new data

APPENDIX I

BASIC MEMORY MAP

000000H: system interrupt vector location
000001H: external interrupt 0 vector location
000002H: external interrupt 1 vector location
000003H: external interrupt 2 vector location
000004H: external interrupt 3 vector location
000005H-00000BH: internal interrupts vector locations
00000CH-00001FH: reserved
000020H-00003BH: trap 0 to trap 27
00003CH-0000BFH: reserved
0000C0H-07FFFFH: SRAM block
080000H-7FFFFFH: expansion board
800000H-80000FH: serial channels
800010H-800027H: external registers
800028H-801FFFH: secondary bus parallel port (fast acces)
802000H-803FFFH: reserved
804000H-805FFFH: secondary bus parallel port (slow access)
806000H-807FFFH: reserved
808000H-8097FFH: internal peripheral bus memory-mapped registers
809800H-809BFFH: internal RAM block 0
809C00H-809FFFH: internal RAM block 1
80A000H-FEFFFFH: expansion board
FF0000H-FFF7FFH: dual-port RAM modules
FFF800H-FFFFFFH: multicast region

REFERENCES

[1] Hunter. I.W., Lafontaine. S.. Hunter. P.. "A scanning laser microscope for muscle fiber studies", Proceedings of the Fourth International Symposium of Optical and Optoelectronic Applied Science and Engineering. 1987.

[2] Hunter. I.W., Lafontaine. S., Nielsen. P.M.F., Hunter, P.J., "An apparatus for muscle cell mechanical testing and laser scanning microscopy", Proc. of the 14th C.M.B.E.C.. Montréal. Québec. Canada. 1988.

[3] Hwang. K., "Advanced parallel processing with supercomputer architectures". Proceedings of the IEEE. Vol. 75. No 10. 1987.

[4] Martel. S., Hunter. I.W.. Nielsen. P.M.F.. Lafontaine. S., Kearney. R.E.. "A parallel supercomputer for biomedical analysis and control", Proc. of the 14th C.M.B.E.C.. Montréal. Québec. Canada. 1988.

[5] Nielsen. P.M.F.. Hunter. I.W.. Lafontaine. S., Martel. S.. "A parallel computer architecture using dual-port memory interconnections", IEEE Transactions on Computers. (1988) (submitted).

[6] "TMS320C30 the third generation of TMS320 family of digital signal processors". Functional Specifications. Rev. 2.1.0. Texas Instruments. 1988.

[7] Shear. D.. "EDN's DSP benchmarks". EDN, Vol. 33. No 20. pp. 126-148. Sept. 29. 1988.

[8] Memory manage~~m~~ent applications handbook, Texas Instruments. 1987.

[9] Leibson. S.H.. "Dynamic-RAM-controller ICs squeeze maximum performance from DRAMs". EDN. Vol. 33. No 17, pp. 91-100. August 18, 1988.

[10] Wilson. R.. "Designers debate advantages of EDACs in small systems". Computer Design, Vol. 27, No 17, pp. 32-33. Sept. 15, 1988.

[11] PAL device data book, Advanced Micro Devices and Monolithic Memories. 1988.

[12] "SN54ALS6300, SN74ALS6300 Input selectable refresh timer", Data Sheets, Texas Instruments. 1987. (Product preview).

[13] Breuninger. R.K.. Schiele. L.. Peprah. J.K.. "System solutions for static column decode". Application Report. Texas Instruments. 1987.

[14] "SN54ALS6310. SN54ALS6311. SN74ALS6310, SN74ALS6311 Static column and page-mode access detectors". Data Sheets. Texas Instruments. 1987. (Product preview).

[15] "SN74ALS6301. SN74ALS6302 Dynamic memory controllers", Data Sheets. Texas Instruments. 1986. (Advance Information).

REFERENCES

[16] "SN74BCT2828 10-bit buffer/driver with series damping resistor", Data Sheets , Texas Instruments.

[17] Memory Data Book. Motorola. 1988.

[18] "SN54AS632. SN54AS634. SN74AS632. SN74AS634 32-bit parallel error detection and correction circuits", Data Sheets. Texas Instruments. 1986.

[19] MacWilliams. F.J.. Sloane. N.J.A.. "The theory of error-correcting codes". North-Holland Mathematical Library. 1986.

[20] Smith. A.J.. "Cache memories". Computing Surveys. Vol. 14, No 3. 1982.

[21] Memory Management Products Design Kits. Texas Instruments. 1987.

[22] Stone. H.S.. "High-performance computer architecture", Addison-Wesley. 1987.

[23] CMOS Data Book. Cypress Semiconductor. 1988.

[24] Hughes. L.. "Chat: an N-party talk facility for the Unix 4.2 operating system", Computer Communications. Vol. 11, No 1. Feb. 1988.

[25] High Performance CMOS Data Book, Integrated Device Technology. 1988.

[26] Sieberman. D.. "Designers hone SBCs to utilize full 32-bit power". Computer Design. Vol. 27, No 15. pp. 50-60. August 15. 1988.

[27] ALS/AS Logic Data Book. Texas Instruments. 1986.

[28] "CS5016 16-bit. 16 us self-calibrating A/D converter", Data Sheets, Crystal Semiconductor Corporation. 1986.

[29] The TTL Data Book. Texas Instruments. 1985.

[30] "PCM56P serial input 16-bit monolithic digital-to-analog converter", Data Sheets. Burr-Brown Corporation. 1987.

[31] The Interface Circuits Data Book, Second Edition, Texas Instruments. 1981.

[32] Wyland, D.C., "Dual-port RAMs simplify communication in computer systems", Application Note AN-02, Integrated Device Technology. 1986.

[33] DRQ3B Parallel DMA I/O Module. User's Guide. Digital. 1986.

[34] Davis. D.B.. "Parallel computers diverge", High Technology, Vol. 7. No 2. pp. 16-22. Feb. 1987.

[35] Hwang. K.. "Advanced parallel supercomputer architectures". Proceedings of the IEEE. Vol. 75. No 10. pp. 1348-1379, Oct. 1987.

[36] Cray Research Inc.. "The Cray 2 computer systems". Technical Brochure. 1985.

[37] ETA Systems. "ETA-10 system overview: Introduction", Technical Note, Publication 1006. Rev. A. Feb. 1986.

REFERENCES

[38] Beetam, J., Denneau, M., Weingarten. D., "The GF11 supercomputer", Proceedings 12th Annual Symposium on Computer Architecture", pp. 108-115, June 1985.

[39] Kuck. D.J., Davidson. E.S., Lawrie. D.H., Sameh, A.H., "Parallel supercomputing today and the Cedar approach", Science. Vol. 231, pp. 967-974. Feb. 1986.

[40] Lubeck. O.. Moore. J.. Mendez, R., "A benchmark comparaison of three supercomputers: Fujitsu VP-200, Hitachi S810/20, and Cray X-MP/2", IEEE Computer. Vol. 18. pp. 10-29, Dec. 1985.

[41] Convex Computer Corporation, "Convex C1 series: XP processors", Richardson, TX, Technical Notes Edition. 1987.

[42] Karplus. W.J.. "Multiprocessors and arrays processors", San Diego. CA, The Society of Computer Simulation, Jan. 1987.

[43] Pfister, G.F.. Brantley. W.C.. George. D..A., Harvey, S.L., Kleinfelder. W.J., McAuliffe. K.P., Melton. E.A.. Norton. V.A., Weiss, J., "The IBM research parallel processor prototype (RP3): Introduction and architecture", Proceedings Int. Conference on Parallel Processing, pp. 764-811, Aug. 1985.