UNIVERSITY OF JORDAN
FACULTY OF GRADUATE STUDIES

DESIGN OF DIGITAL SIGNAL PROCESSING BOARD USING TMS32020 MICROPROCESSOR

BY

NIDAL SAMIH AL-DALI

SUPERVISED BY

Prof. Dr. ISAM ZABALAWI
&
Dr. BASSAM KAHALEH

A Thesis
Submitted in Partial Fulfilment of the Requirements for the Degree of Master of Science in Electrical Engineering Communications, Faculty of Graduate Studies, University of Jordan.

January, 1993

EXAMINING COMMITTEE

CHAIRMAN:

Prof. Isam Zabalawi _____

MEMBER:

Prof. Mohamed K.Abdelazeez m. K. abdel _____

MEMBER:

Dr. Andrawos Sweidan _____

MEMBER:

Dr. Souheil Odeh Souheil Odeh _____

ACKNOWLEDGMENT

I would like to express my sincere thanks and appreciation to my supervisors Prof. Isam Zabalawi and Dr. Bassam Kahaleh for there help and valuable advice during the course of this study.

Sincerest gratitude to prof. Mohamed K. Abdelazeez, Dr. Andrawos Sweidan and Dr. Suoheil Odeh for their participation and corporation as committee members.

The author extends his special thanks to engr. Ali Al-Mousa for his participation in organization the layout of the board, also a great appreciation is extended to Engineer Gheith Abandah for his support and sincere efforts.

I gratefully acknowledge the efforts provided by the staff of Electrical Engineering Laboratories for their help and assistance.

Finally, the deepest appreciation and sincerest gratitude is extended to my father, mother and my wife for their patient and love.

CONTENTS

LIST OF FIGURES & TABLES:-

ABSTRACT

This project is aimed toward building a low cost general purpose digital signal processing board using TMS32020 microprocessor as the heart of the project. This microprocessor and his family TMS are used extensively for such applications, it is produced by Texas Instruments in 1985. A complete system was designed and built in the Electrical Engineering Department laboratories using computer programs taking into consideration cost, high efficiency and, flexible design. This project will be the first step in building a digital signal processing laboratory in this department and it will be used in applications such as digital filters, acoustic systems, producing and analyzing digital signals and many other applications. This thesis will present all practical and theoretical aspects and programs used with some applications.

خلاصة

ان هذا المشروع يهدف الى بناء لوحة معالجة اشارات رقمية متعددة الاغراض وبتكاليف قليلة، وتم استخدام المعالج TMS32020 كقلب لهذا المشروع، ويستخدم هذا المعالج على نطاق واسع مع عائلته من TMS لمثل هذه التطبيقات، وقد تم انتاج هذا المعالج بواسطة شركة Texas Instruments الامريكية الشهيرة في العام ١٩٨٥.وفي هذا المشروع تم تصميم نظام معالجة الاشارات الرقمية بحيث يأخذ بعين الاعتبار الكلفة والكفاءة والمرونة في التشغيل، ولقد تم التصميم بإستخدام برامج الحاسوب المتطورة، وثم تنفيذه في مختبرات قسم الهندسة الكهربائية.إن بناء هذه اللوحة "لوحة معالجة الاشارات الرقمية" سيكون اللبنة الاولية لبناء مختبرات الاشارات الرقمية في القسم، وسيكون من الممكن استخدام هذه اللوحة في تطبيقات المرشحات الرقمية والانظمة الصوتية وتحليل وانتاج الاشارات الرقمية وتطبيقات عديده اخرى.وفي هذا البحث سيتم تقديم كافة جوانب المشروع العملية والنظرية والبرامج المستخدمة فيه و تطبيقات عملية.

CHAPTERT 1

INTRODUCTION

In the 80's, Digital Signal Processing has made a great progress in both theoretical and practical aspects of the field, while more DSP algorithms are being developed, better tools are also being used and developed to implement these algorithms. Such applications tend to require high speed microprocessors and systems[1].

Digital Signal Processors are essentially high speed microprocessors, designed specifically to perform computational intensive operation, normally required in implementing digital signal processing algorithms. By taking advantage of advanced computer architectures[2], parallel processing, and dedicated DSP instruction sets, the new generation of processors can execute millions of DSP operations per second. In addition, new technologies allow complicated DSP Algorithms to be implemented in a tiny silicon chip, which previously required the use of a minicomputer and an array processor. Because of these and many other advantages such as reliability, reproducibility, compactness and efficiency, digital signal processors are becoming more prevalent in areas of general purpose digital signal processing, telecommunications.voice, speech,instrumentation,

graphic, imaging, control, and military.[3]

This project is based on a digital signal processor from the TMS320 Family. This family contains several processors including the first MOS microprocessor capable of executing five million instructions per second. This was achieved through the use of comprehensive and efficient, instruction set, and a highly pipe lined architecture.[4] Special instructions, such as multiply /accumulate with fast data move increase the performance of most DSP programs. Also a comprehensive set of instructions simplifies programming application software.[5]

The TMS320 processor utilizes a modified Harvard architecture for speed and flexibility. The program and data memories lie in two separate address spaces, permitting a full overlap between instruction fetching and execution. It allows transfer between program and data Locations to increase flexibility, and maximizes processing performance by maintaining two separate buses for code and data.

Texas instruments' TMS320 Family consists of many generations of digital signal processors. The first generation contains the TMS32010, TMS320C10, TMS32011 and TMS32010-25,[6] while the second generation consist of the type TMS32020.[7]

The features mentioned earlier are common among all processors in the family. Some specific features are added in each processor to provide different cost/performance tradeoffs.[8]

The selection of the TMS32020 processor for this project was made after considering the intended use of its availability, the powerful instruction set. The TMS32020 digital signal processor is the second generation member of the TMS320 family of VLSI processors. Its architecture is based upon that of the TMS32010, the first member of the TMS320 family. The TMS32020 provides a greatly enhanced memory address space, and on-chip memory of 544 words of data and program.

The TMS32020 has an increased throughput accomplished by implementing a single cycle multiply/accumulate instruction with optional data movement. It includes five auxiliary registers with a dedicated arithmetic unit, and implements a faster I/O throughput for data intensive applications. TMS32020 has a comprehensive instruction set of 109 instructions to facilitate software development. It contains special "Repeat" instruction for streamlining program space and execution time. This instruction allows the execution of the next single instruction (N+1) times, where N is defined by

an 8-bit repeat counter. The instruction being repeated is fetched only once. So many multicycle instructions become one or two cycles when repeated.

The architectural design of the TMS32020 emphasizes overall system speed, communication, and flexibility in processor configuration. Special instructions provide block memory transfer, communication to slower off-chip devices multi processing implementation, and floating-point support. Peripherals such as a timer and a serial port, have been integrated on-chip to reduce overall system cost.

The TMS32020 is fabricated in 2.4μ NMOS technology and has a chip area of 119k square mil. It is produced in 68-pin grid array package and has a typical power consumption of 1.2W, The maximum clock frequency is 20.5 MHz for an execution rate of 5 million instructions per second.[9]

CHAPTER 2

HARDWARE

2.1 DSP System Requirements:

The DSP system is designed to meet the requirements of the Electrical Engineering Department in the University of Jordan to establish a digital signal processing laboratory. This work can be used in defferentapplications as,i.e. digital filtering, Data Acquisition, storage oscilloscope, ...etc. The DSP system is designed as a general purpose digital Signal Processor with the following specifications:

1- TMS32020 Digital signal Processor as the CPU, operating at 20 MHz.

2- One analog input channel for signals up to 20KHz and maximum amplitude in the range of ∓ 2.5 Volts.

3- One Analog output channel with maximum amplitude in the range of ∓ 10 Volts.

4- Data memory (RAM) of 64k*16 for data storage.

5- Program memory (RAM) of 32k*16 for program storage.

6- Program memory (EPROM) of 4k*16 for firmware storage and system operation.

7- Serial Port interface (RS232) for communication with the personal computer (PC).

Thus, the system is intended to be a portable DSP system that can be used with any PC. The PC is used for down Loading the user program to the DSP system memory and other operations concerning data transfer between the two systems.

The system is designed with all necessary components on one single printed circuit, and requires only an external ∓15V power supply with 3 ampere output. The user has a RAM space of 32k*16 available for downloading this program. The EPROM is used to contain the operating program of the system including the communication protocol used in conjunction with the PC. A block diagram of the system is shown in figure 2.1 and the details od these blocks as follows:

2.2. THE MICROPROCESSOR[10]

The TMS32020 Digital Signal Processor is used as the CPU of the system. The key features of this chip are:

- 200-ns instruction cycle time.
- 16-bit instruction words.
- 16-bit data words with internal 32-bit operations.
- 544 words of on-chip data RAM of which 256 words are programmed as either data or program memory.
- 128k words of data/program space (64k x 16 words of data memory, 64k x 16 words of program memory).

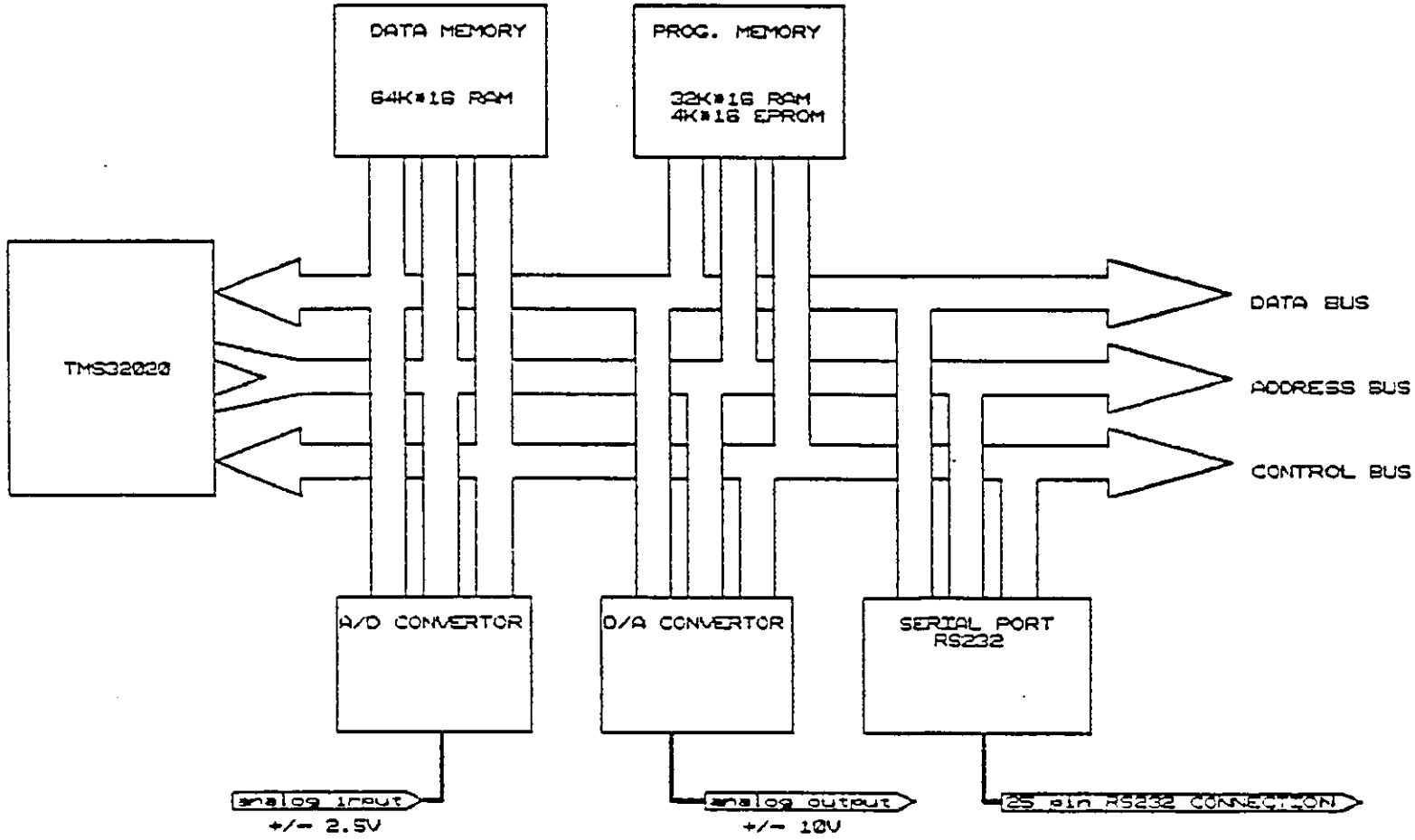


Figure 2.1 : Block diagram of the system.

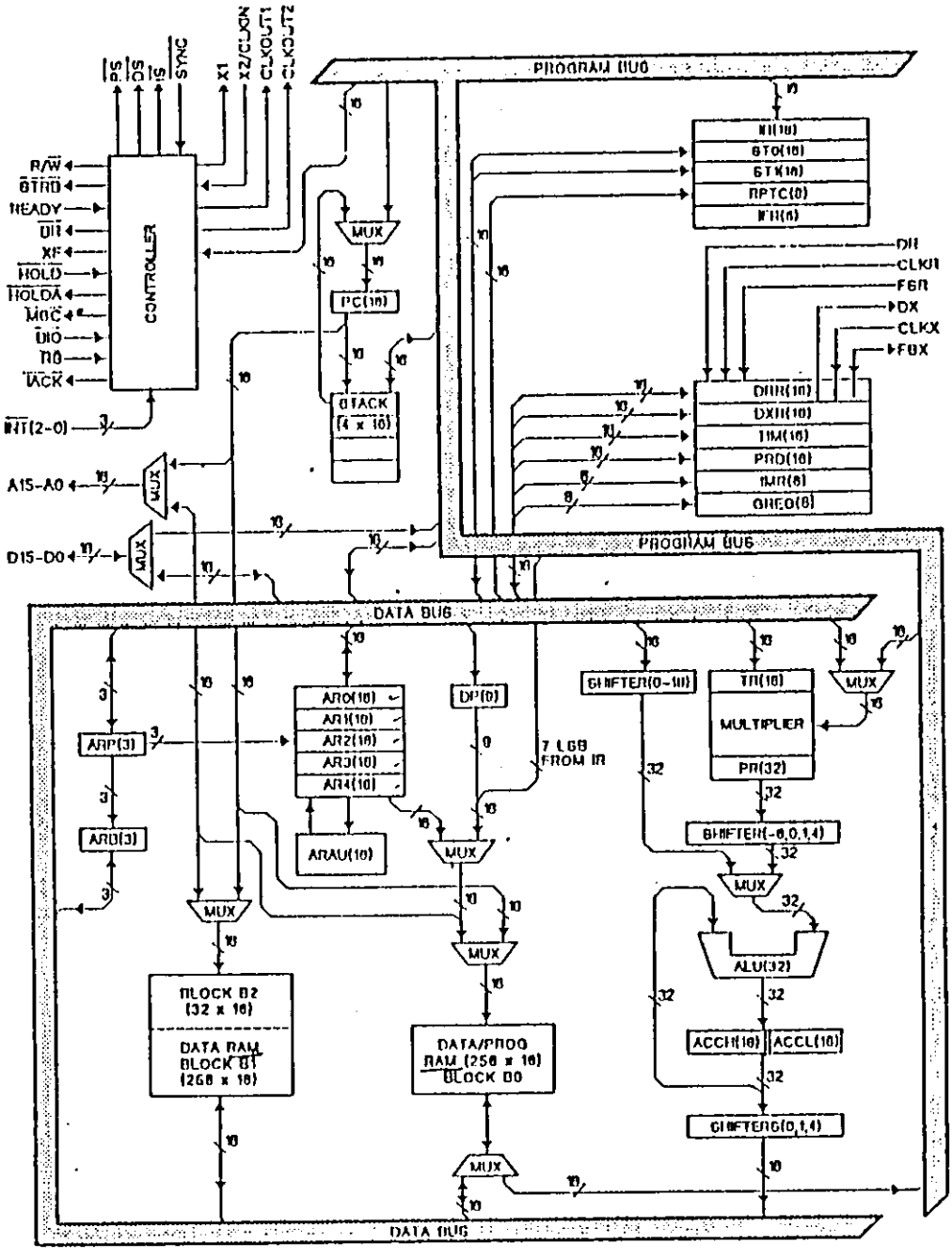
- Sixteen input and sixteen output channels.
- Directly accessible external data memory space.
- Global data memory interface.
- 32-bit ALU and accumulator.
- Single-cycle multiply/accumulate instructions.
- 0 to 16-bit scaling shifter.
- Fractional/integer arithmetic operations.
- Bit manipulation and logical instructions.
- Instruction set support for floating-point operations.
- Block moves for data/program management.
- "Repeat" instructions for increasing the efficiency of program space utilization while allowing pipelined operation.
- Five auxiliary registers for indirect addressing with auto-increment/decrement capability and temporary storage.
- Dedicated arithmetic unit for operating on the auxiliary registers.
- Serial port for supporting multi processing environment, serial analog-to digital converters, ...etc.
- Synchronization input for synchronous multi processor configurations
- Wait states for communication with slower off-chip memories and/or peripherals.

- On-chip timer for real time control operations
- Three external, maskable, user interrupts
- Input pin polled by software branch instruction
- Output pin for signaling external devices (set/reset by instructions)
- 2.4-micron NMOS technology
- Single 5-V supply
- On-chip clock generator

The functional block diagram of the TMS32020 is shown in fig.(2.2) It outlines the principle blocks and data paths within the processor, and shows all of the TMS32020 interface pins. The TMS32020 implements internally functions that other processors typically provide through software or microcode, such as the single cycle 16x16 bit multiplication.

2.2.1 The Memory Unit[9]:

The TMS32020 provides a total of 544 16-bit words of on-chip data RAM, of which 288 words are always data memory and 256 words are used as either program or data memory. The 544 words are organized as three separate memory blocks: B0, B1 and B2. Block B0 contains the 256 words that are programmable as either data or program memory. Blocks B1 and B2 contain the other 288 words which are always used as data memory.



NOTE:

ACCH= Accumulator high
ACCL = Accumulator low
AHAU= Auxiliary register arithmetic unit
ARB = Auxiliary register pointer buffer
ARP = Auxiliary register pointer
DP = Data memory page pointer
DRR = Serial port data receive register
DXR = Serial port data transmit register
GREG= Global memory allocation register
IFR = Interrupt flag register
IMR = Interrupt mask register
IR = Instruction register
PR = P register
PRD = Period register for timer
TR = T register
TIM = Timer
ST0,ST1 = Status registers
RPTC = Repeat instruction counter

Figure 2.2 :Block Diagram of TMS32020 Digital Signal Processor [9]

The TMS32020 runs at full speed when operating through the on-chip program RAM or a fast external program memory. One can use slower, less-expensive external memory by using the available READY line to inject wait states.The latter approach was selected in the design of this DSP project.

2.2.2 Memory Maps[9]

The TMS32020 provides three seperate address spaces for program memory, data memory, and I/O as shown in fig.(2.3).The type of generated address is indicated externally by means of the $\overline{PS}$, $\overline{DS}$, and $\overline{IS}$ signals.

The data memory space is devided into 512 pages, each page consists of 128 × 16 byte resulting a 64k × 16 space, an 8-bit data memory page pointer points to one of the 512 page in the direct addressing mode.

The on-chip memory blocks B0, B1 and B2 comprises a total of 544 words of RAM. The program/data RAM block B0 (256 words) resides in pages 4 and 5 of the data memory map when configured as data RAM, and in pages 510, 512 when configured as program RAM. The CNFD/CNFP instructions are used to configure block B0 as either data or program memory.

Block B1 (data RAM) resides in pages 6 and 7 while block B2 resides in the upper 32 word of page 0. The remaining part

MEMORY MAPS (1) A

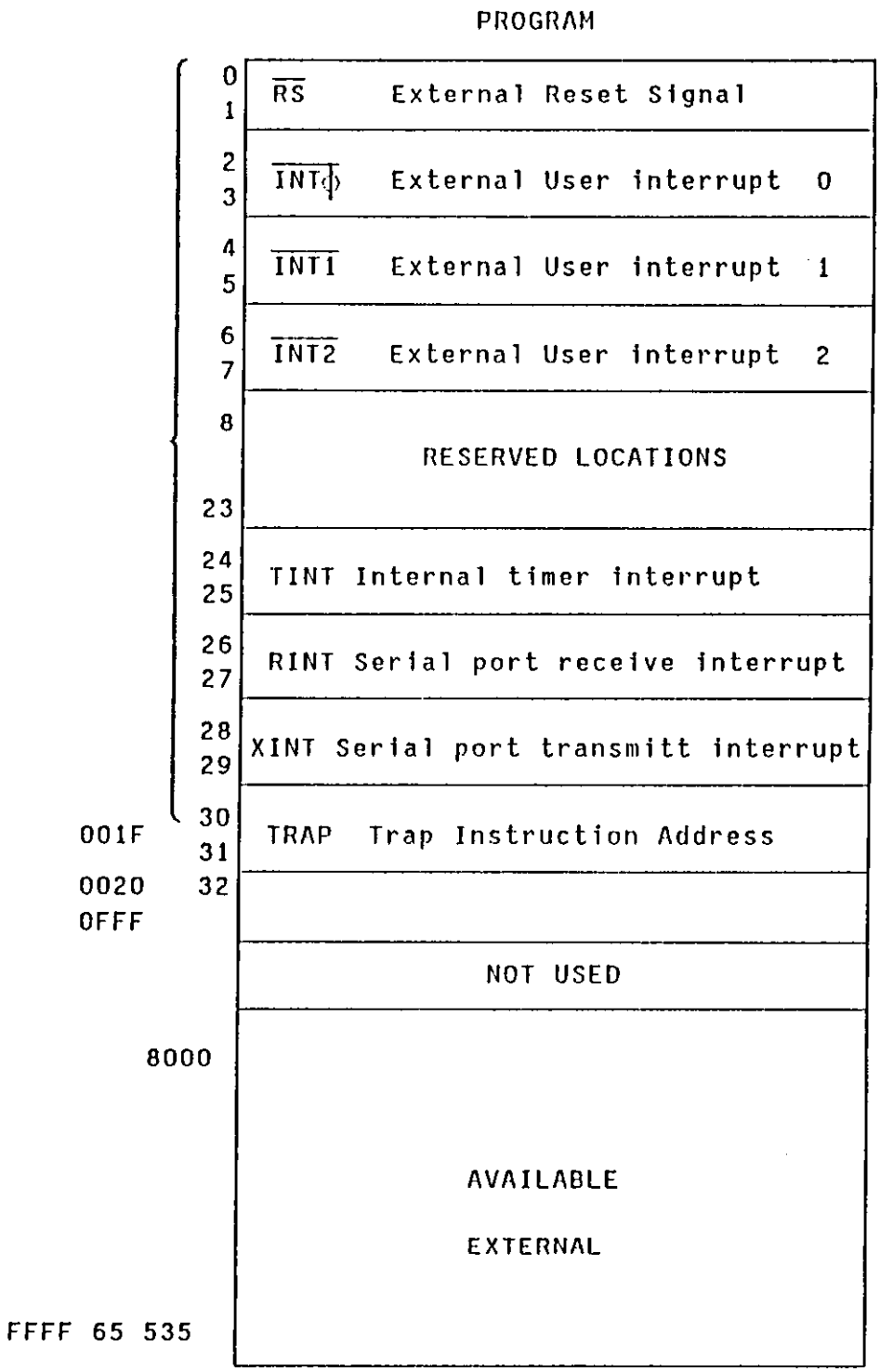


Fig (2.3)a (A) Address Maps After A CNFD Instruction.

MEMORY MAPS (1) B

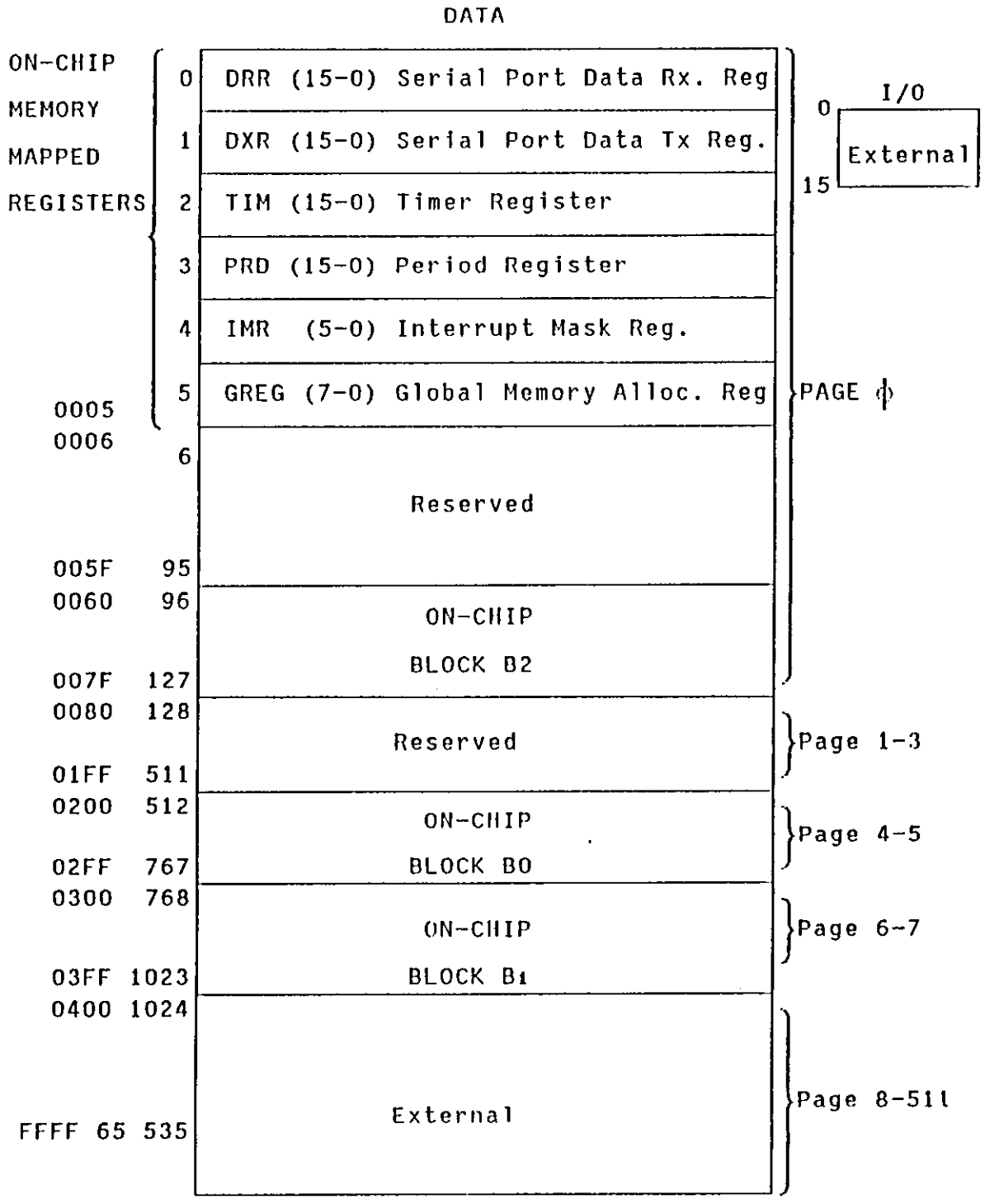


Fig (2.3)a (B) Address Maps After A CNFD Instruction.

MEMORY MAPS (2) A PROGRAM

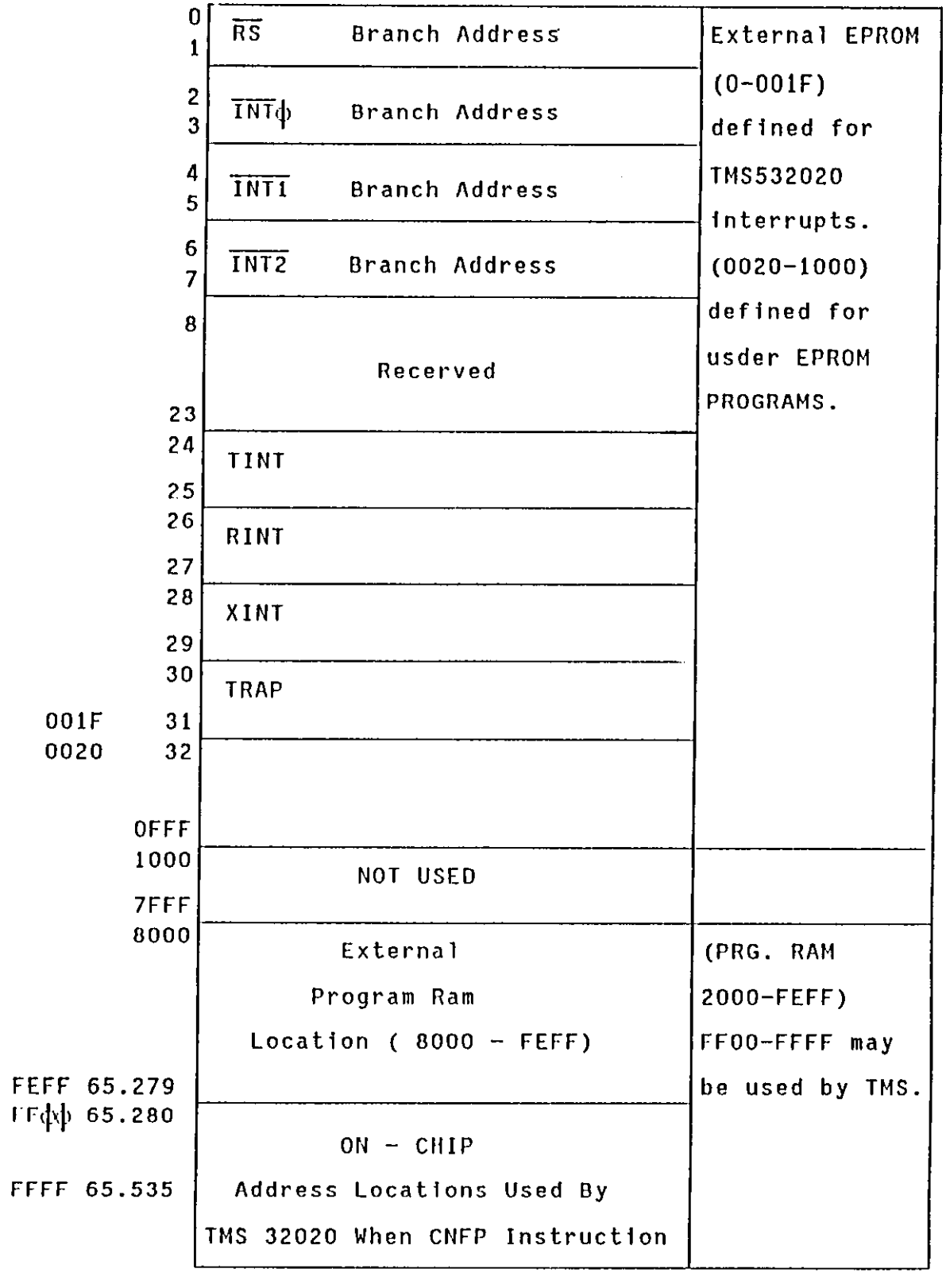


Fig (2.3)b (A) Address Maps After A CNFP Instruction

MEMORY MAPS (2) B

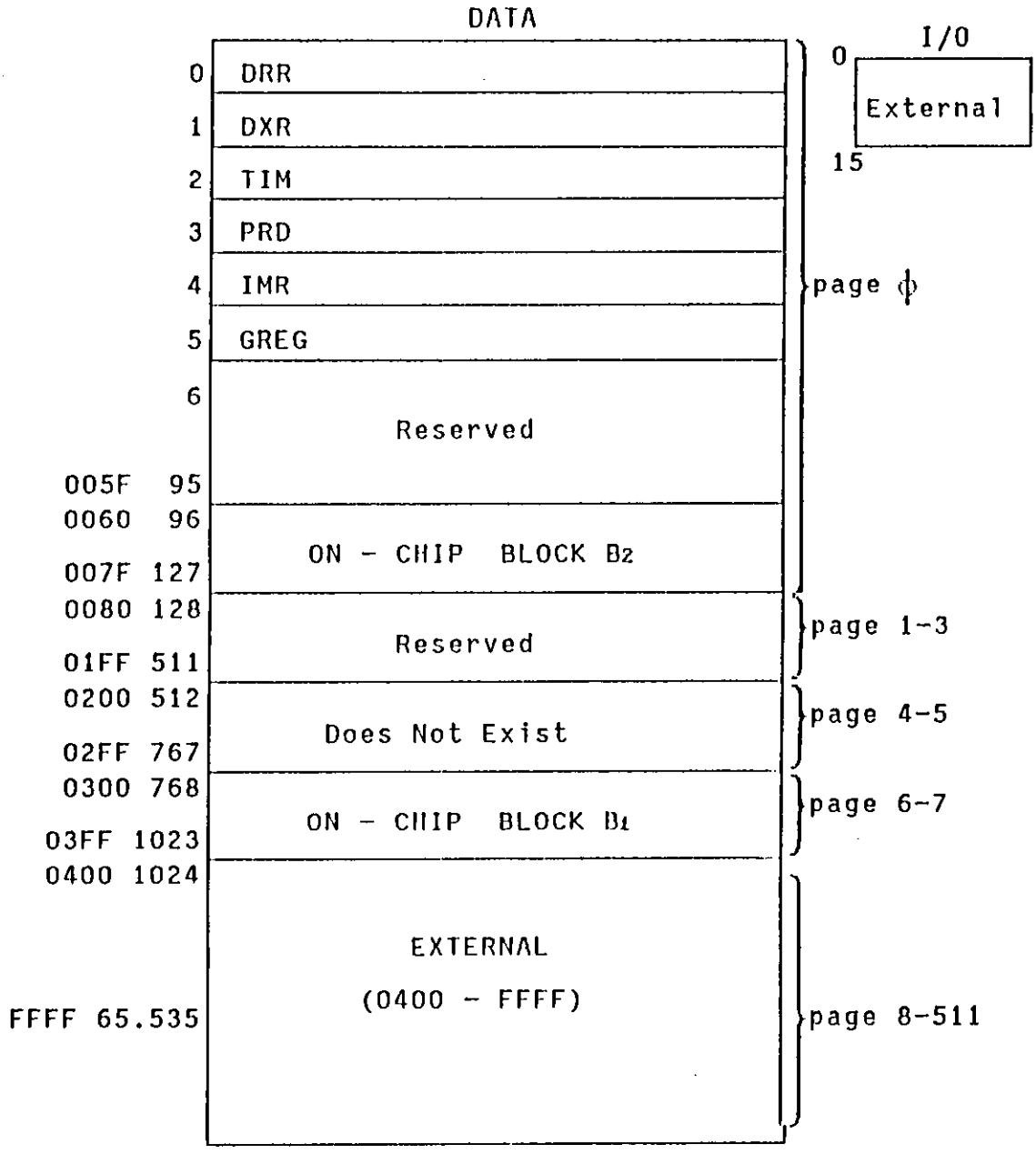


Fig (2.3)b (B) Address Maps After A CNFP Instruction

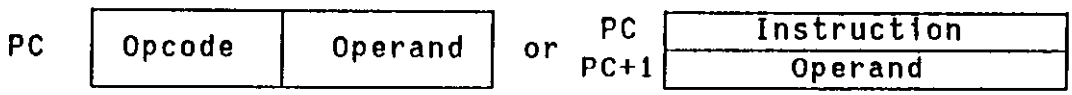
of the data memory map is composed of memory mapped registers and reserved locations, reserved locations may not be used for storage and their contents are undefined when used.

2.2.3 Memory Addressing Modes[9]

The TMS32020 can address a total of 64k words of program memory and 64k words of data memory. Addressing data memory can be done by one of the following three ways:-

1) Immediate addressing mode:

In this mode the data is based on a portion of the instruction word(s), i.e the instruction contains the value of the immediate operand.



2) Direct addressing mode:

In this mode, the least seven bits of the instruction represents the location of the data memory address within one page (128 words), the page is defined by the 9-bit data memory page pointer (DP), which points to one of 512 possible pages. In other word, the 16-bit data address consist of 9-bits from the data memory page pointer and 7-bits from the instruction itself.

3) Indirect addressing mode:

In this mode, any location in the 64k data memory space is accessed via the 16-bit address contained in the auxiliary register, the auxiliary register pointer (ARP) refers to one of five auxiliary registers (AR0-AR4).

2.2.4 Memory-Mapped Registers[9]

The TMS32020 has 6 registers that are mapped into the data memory space. These memory-mapped registers may be accessed in the same manner as any other data memory location with the exception that block moves using BLKD are not allowed from the memory-mapped registers. Table (2.1) Lists the names of memory mapped registers.

Register name	Address Location	Definition
DRR(15-0)	0	Serial port data receive register
DXR(15-0)	1	Serial port data transmit register
TIM(15-0)	2	Timer register
PRD(15-0)	3	Period register
IMR(5 -0)	4	Interrupt mask register
GREG(7-0)	5	Global memory allocation register

Table (2.1)

2.1.5 Auxiliary Registers

The TMS32020 provides a register file containing five auxiliary registers (AR0 - AR4). The auxiliary registers are used for two purposes: for indirect addressing the data memory, and for temporary data storage.

Register indirect addressing allows referencing memory by

placing the data memory address in one of the five available auxiliary registers. These registers are pointed to by a three-bit Auxiliary register pointer (ARP) that is loaded with 0, 1, 2, 3, or 4 for AR0 - AR4.The auxiliary registers and the ARP may be loaded either directly from data memory or by an immediate operand specified in the instruction.

The auxiliary register file (AR0 - AR4) is connected to an auxiliary register Arithmetic unit(ARAU), The ARAU is used to auto-index the current auxiliary register while the data memory location is being addressed, the ARAU performs the following functions:

AR(ARP) + AR0 ——→ AR(ARP) Index the current AR by adding a 16-bit integer contained in AR0.

AR(ARP) - AR0 ——→ AR(ARP) Index the current AR by subtracting a 16-bit integer contained in AR0.

AR(ARP) + 1 ——→ AR(ARP) Auto increment the current AR by one.

AR(ARP) - 1 ——→ AR(ARP) Auto decrement the current AR by one.

AR(ARP) ——→ AR(ARP) AR(ARP) in unchanged.

2.2.6 Program Counter and Stack

The TMS32020 contains a 16-bit program counter (PC) and

four-location stack memory.

The stack is used for temporary storage during interrupts, calls, and returns.

2.2.7 Scaling Shifter

The TMS32020 scaling shifter has a 16-bit input connected to the data bus and 32 bit output connected to the ALU. The scaling shifter is capable of shifting an operand to the left 0 to 15 bits. The LSBs of the output are filled with zeros. The MSBs may be either filled with zeros or sign-extended depending upon the status programmed into the SXM (sign extension mode) bit of status register ST0.

2.2.8 Arithmatic Logic Unit and Accumulator

The TMS32020 32-bit Arithmetic logic unit (ALU) and accumulator perform a wide range of arithmetic and logical operations. The 32 bit Accumulator is split into two 16-bit segments for storage in data memory. They are referred to as ACCH (accumulator high) and ACCL (accumulator low.)

2.2.9 Multiplier T & P Registers

The TMS32020 has a two's complement 16x16 bit hardware multiplier capable of producing a 32-bit product in a single instruction cycle time. The following registers are associated with this multiplier unit:

* A 16-bit Temporary Register (TR) that holds one of the two operands to be multiplied.
* The other operand can be either an immediate operand or the contents of the addressed data memory location specified by the instruction.
* A 32-bit Product Register (PR) that holds the product.

When multiplying two 16-bit operands, in two's complement code, the 32-bit product is loaded in the 32-bit product register. This PR may then be transferred to the ALU directly, or optionally through the shifter to be shifted before reaching the ALU.

2.2.10 System Control

Control operations are provided on the TMS32020 by an on-chip timer, repeat counter, external and internal interrupts and an external reset signal, these operations are explained in the following sessions:

2.2.10.1. TIMER

The TMS32020 provides a memory mapped 16-bit timer for control operation. The on-chip timer register (TIM) is a down counter that is continuously clocked by the internal clock. The clock is derived from clockout1 by dividing the frequency

by 4. The RESET input sets the timer to its maximum value FFFF. A timer interrupt TINT is generated every time the timer reaches zero. Therefore, if timer is not used, the timer interrupt should be masked or disabled by a DINT instruction, which disables all maskable interrupts.

2.2.10.2. REPEAT COUNTER

The TMS32020 design includes a repeat feature that allows a single instruction to be performed up to 256 times. This feature can be used with instructions such as multiply/ accumulate, block moves, I/O transfers, and table read/writes. Thus, these instructions becomes effectively single cycle instructions.

2.2.10.3. RESET

The use of $\overline{RS}$ signal asynchronously causes the TMS32020 to terminate execution and forces the program counter to zero. program execution begins at location 0 which contains a branch instruction to direct program execution to a suitable system initialization routine. This external Reset signal is generated by pressing a little push button on the board.

2.2.11. Status Registers

The TMS32020 has two status registers, ST0 and ST1 that contain the status of various conditions and modes.

2.2.12. Interrupts[9]

The TMS32020 has three external maskable interrupts $\overline{INT2}$, $\overline{INT1}$, $\overline{INT0}$ available for external devices to interrupt the processor. Internal interrupts may be generated by the serial port (RINT & XINT), the timer (TINT), or the software interrupt instruction (TRAP). Interrupts are prioritized, with reset as the highest priority and the serial port's transmit interrupt as the lowest priority.

A built-in mechanism protects the execution of multicycle instructions from interrupts. If an interrupt occurs during the execution of a multicycle instruction the interrupt is not processed until the instruction is completed.

For proper interrupt handling, $\overline{INT2}$-$\overline{INT0}$ must be syncronized with the TMS32020, an external hardware shown in fig. (2.4) is suggested by the TMS32020 data cheet and used in the design of the board, the interrupt signal is syncronized with the rising edge of clockout1 signal of the TMS32020.

$\overline{INT0}$ is for RxReady in 8251 A

$\overline{INT1}$ is for TxReady in 8251 A

$\overline{INT2}$ is for A/D 7580

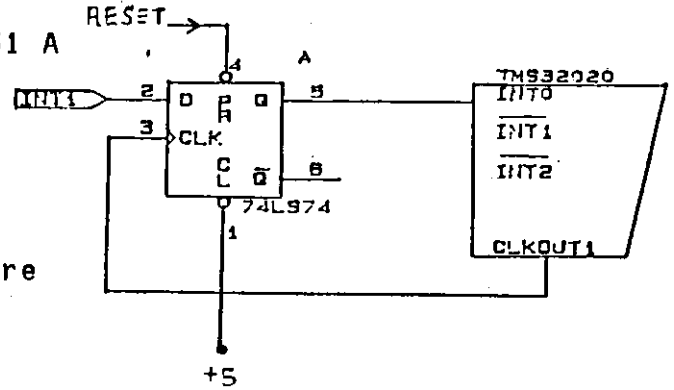


fig. (2.4): External hardware for proper interrupt.

2.3. MEMORY[9]

Considering the intended applications for this DSP project two types of memories are required. Data memory would be used to store the required various data for applications such as the coefficients of digital filters, data samples for digital storage oscilloscope ...etc. Program memory would be used to store various Programs to run the required applications. A memory arrangement was designed as follows:-

1- DATA memory of (64k * 16) words, using the static RAM IC number 65256-15, with 150ns maximum access time. Four IC's were used, each one providing 32k bytes, for a total of 64k*16. Interfacing the TMS32020 with such memories requires the use of one wait state as shown in table (2.2). This was easily accomplished by connecting the READY input to microstate complete $\overline{MSC}$ signal, both on the TMS32020 IC. The DATA memory address space covers the range of 0400 - FFFF.

#WAIT STATES	MEMORY ADDRESS ACCESS TIME
0	85 ns
1	285 ns
2	485 ns
. . .	. . .
n	85 ns+(200 ns)*n

Table (2.2):- No. of wait states

2- PROGRAM memory of (32k * 16) using the same 65256-15 static RAM IC's. Two IC's are used to provide 32k*16.

This size of memory should be more than enough for most DSP applications. The program memory address space covers the range of 8000 - FEFF.

3- EPROM Program memory of (4k * 16) words using 2732A-20 IC, which contains 4k bytes of erasable and electrically programmable ROM. Two IC's, are used to provide 4k*16 configuration. This EPROM has an access time of 200ns and therefore is in the range of one wait state requirement.

The EPROM is used to store the operating program (firmware) of the system.Its address space covers the range of 0000 - 1000.

When interfacing the TMS32020 processor to the RAMs & EPROMs, buffer IC's are required to drive the address and data buses. The 74LS245 bidirectional buffers were used. The direction is selected by the R/$\overline{W}$ signal of the TMS32020 IC.

The chip select ($\overline{CE}$) inputs of the memory ICs are driven by program select $\overline{PS}$, the data select $\overline{DS}$ and the A15 signals of the TMS32020 IC through a 2-to-4 decoder IC, as shown in the schematic diagram in appendix (.1.).

2.3. ANALOG INPUT CHANNEL

The analog input channel connects the external analog world to the DSP system. It is designed to suit analog signals, with a maximum of 20KHz frequency. The input signal is taken through a sample & Hold and an A/D converter before it is captured by the CPU and then processed according to the required application.

Due to the wide variations among A/D converters especially with cost a compromise between the cost, accuracy, and sampling rate was made with the selection of the AD7580 IC.[11] It is manufactured by Analog Devices, with the following main specifications:-

* 10 bit sampling A/D converter
* 20μs conversion time
* ON-chip sample-Hold
* 50 KHz Sampling rate
* 25 KHz Full power input bandwidth
* choice of Data format.

The input signal is required be in the range of $\mp$ 2.5 Volts when sampling bipolar signals. The step size in this case would be 4.88mv/bit. Figure (2.5)shows a suitable input circuit suggested by the manufacturer data sheets.

Vin(t)		OUTPUT CODE
- 2.500	V	00 0000 0000
- 2.49512	V	00 0000 0001
0.00	V	10 0000 0000
+ 2.49512	V	11 1111 1111

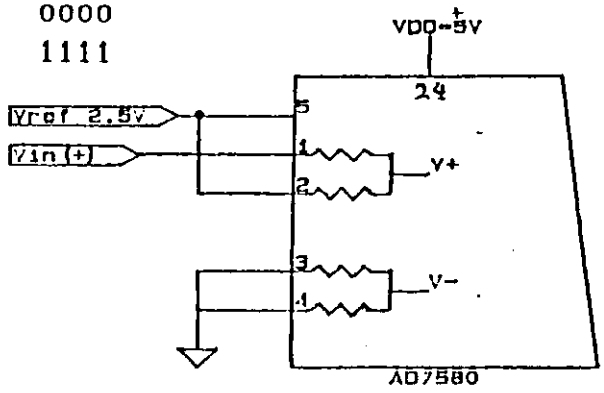


Figure (2.5):- The input circuit.

The circuit shown is for the single-Ended Bipolar Operation mode (-2.5v to +2.5v), which is used in this design.

The AD7580 requires an external voltage reference of +2.5V. Figure (2.6) shows a suitable 2.5V voltage reference circuit using the LM336 IC.

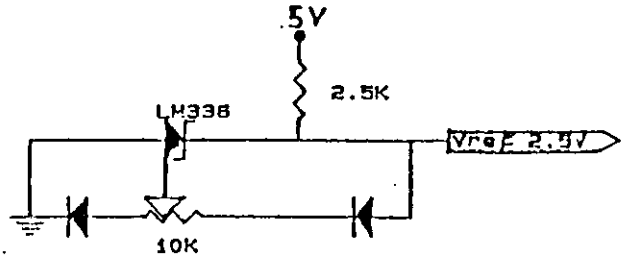


Figure (2.6):-
2.5v voltage reference.

The +2.5V output is calibrated by adjusting the 10K multiturn potentiometer.

The A/D input signal (pin 1) requires a protection circuit to protect the internal IC from undesirable over voltage conditions. In addition, a simple RC low pass filter is used with a cutoff frequency around 25 KHz as shown in figure (2.7).

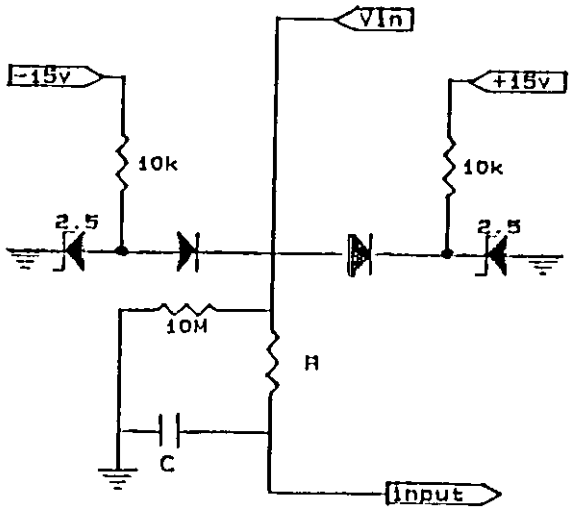


figure (2.7) a simple RC low pass filter

The A/D chip is selected ($\overline{CE}$) when the TMS32020 references port 2.

The $\overline{INT}$ output of AD7580 is connected to $\overline{INT2}$ of TMS32020 the processor when the conversion is complete in order to read the digital bits after conversion.

RDY output of the AD7580 is connected to D15 of the TMS32020, this RDY signal is accessed during Read Cycle. When asserted, it is low during conversion and high impedence when conversion is complete. The user can check D15 to insure that the conversion is complete and then read the digital bits. The user could know that the conversion is complete without enabling the interrupt.

2.5. ANALOG OUTPUT CHANNEL

The analog output channel allows the DSP system to output analog signals via an D/A converter. An excellent D/A converter was chosen from BURR-BROWN, type DAC 707 JP with the following specifications[12]:-

1- microprocessor compatible with write, clear, control lines, and latch enable lines.

2- High accuracy with 16-bit resolution.

3- linearity error = $\mp$ 0.003% of FSR max.

4- Voltage output type with output voltage amplifier

5- Input coding in Binary two's complement (BTC) and compatible with TTL, LS-TTL, HC.

6- Output voltage ∓ 10V with min ∓ 5mA.

The DAC707 JP requires a power supply V_{DD} = + 5V and + V_{CC} = 15V and - V_{CC} = - 15V. The DAC707 JP also has an offset and gain adjust capabilities by trim potentiometers.

2.6. SERIAL PORT RS232

As previously mentioned, the DSP system is designed to be a portable system[13]. It can be connected to any personal computer that has a standard RS232 communication port. Therefore, a standard RS232 communication port was designed with full handshaking protocol to be more general.

Intel's programmable communication interface, 8251A chip, was used as USART (Universal synchronous/Asyncronous & Receiver Transmitter) for data communicat i ons between the DSP system and the PC. It has the following features:-

* synchronous and Asynchronous operation.
* Asynchronous 5-8 bit character length.
* Programmable clock rate x1, x16, or x64 times Baud rate.
* Break character generation, 1, 1½ or 2 stop bits.
* False start bit detection.
* Automatic Break detect and handling.

* Asynchronous Baud rate DC to 19.2k Baud.

* Full-Duplex, Double-Buffered transmitter and receiver.

* Error Detection-Parity, Overrun and Framing.

* All inputs and outputs are TTL compatible.

The USART accepts data characters from the CPU in Parallel format and then converts them into a continuous serial data stream for transmission. Simultaneously, it can receive serial data streams and convert them into parallel data characters for the CPU.

The USART signals the CPU whenever it is ready to accept a new character for transmission or whenever it has finished the assembly of a received character. The CPU can read the status of the USART at any time.

A baud rate generator, IC 4702 B[14], is used to provide the required clock signals for data transmission sub-system. It can be programmed by 4-dip switches to generate 14 commonly used baud rates using the on-chip crystal oscillator circuit. For this purpose, a standard 2.4576 MHz crystal is used.

The clock output Pin of the 4702 B IC[14], which gives 2.4576 MHz is used to clock the 8251 A and the AD7580 ICs. Table (2.3) shows the truth table for baud rate select inputs.

Table (2.3):- Truth table for baud rate select inputs (5)

S_3	S_2	S_1	S_0	Baud rate output	Actual output frequency (H_z)
L	L	L	L	Multiplexed input (I_m)	
L	L	L	H	Multiplexed input (I_m)	
L	L	H	L	50	800
L	L	H	H	75	1200
L	H	L	L	134.5	2152
L	H	L	H	200	3200
L	H	H	L	600	9600
L	H	H	H	2400	38400
H	L	L	L	9600	153600
H	L	L	H	4800	76800
H	L	H	L	1800	2880
H	L	H	H	1200	19200
H	H	L	L	2400	38400
H	H	L	H	300	4800
H	H	H	L	150	2400
H	H	H	H	110	1760

The baud rate output pin is connected to the receiving & transmitting clock of the 8251A IC. The USART 8251A can interrupt the CPU in both transmitting and receiving modes, in the following way:-

Transmitter ready (TxRDY) output signals the CPU that the transmitter is ready to accept a data character, the TXRDY output pin is used as an interrupt to the CPU via $\overline{INTI}$ input. Alternatively for polled operation, the CPU can check TxRDY using a status read operation. TxRDY is automatically reset by the leading edge of $\overline{WR}$ signal when a data character is loaded from the CPU.

Receiver ready (RxRDY) output indicates that the 8251A IC contains a character that is ready to be read by the CPU. RxRDY output pin is used to interrupt the CPU via $\overline{INTO}$ input. Alternatively for polled operation, the CPU can check the RxRDY signal using a status read operation.

The chip-select of 8251A is activated by asserting address 0000 or 0001 on the address bus. If the address is 0000 then the 8251A knows that there is a word either read or write by processor, the address 0001 is used to inform 8251 A that these is a control or status write or read operation respectively, i.e 0000 is used for data read or write, 0001 is used for control word or status information.

The communication signals that are connected to a standard 25 Pin DB25 connector are: TXD, RXD, $\overline{DTR}$, $\overline{DSR}$, $\overline{RTS}$, $\overline{CTS}$ and GND. These signals are connected from the 8251 A to the 25 Pin connector via line drivers and receivers IC's 1488 and 1489 respectively. The R/$\overline{W}$ signal from the microprocessor is decoded into two signals $\overline{RD}$ & $\overline{WR}$ by the following circuit of figure (2.8)..

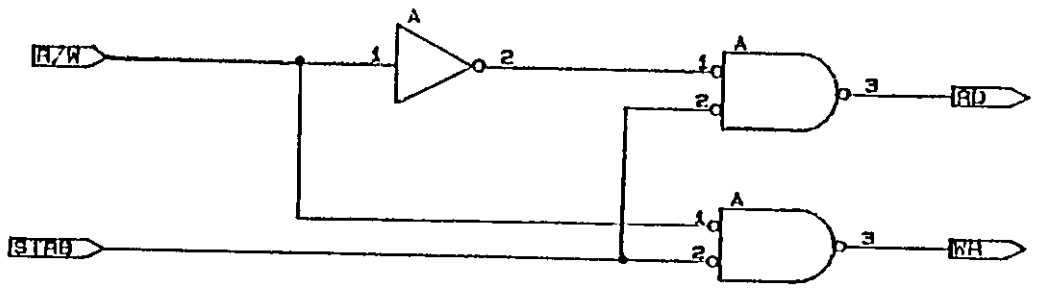


Figure (2.8) The R/$\overline{W}$ signal is decoded into two signals $\overline{RD}$ & $\overline{WR}$

The Address decoding system is designed with three 2-to-4 decoders type 74LS139. Figure (2.9) shows the circuit.

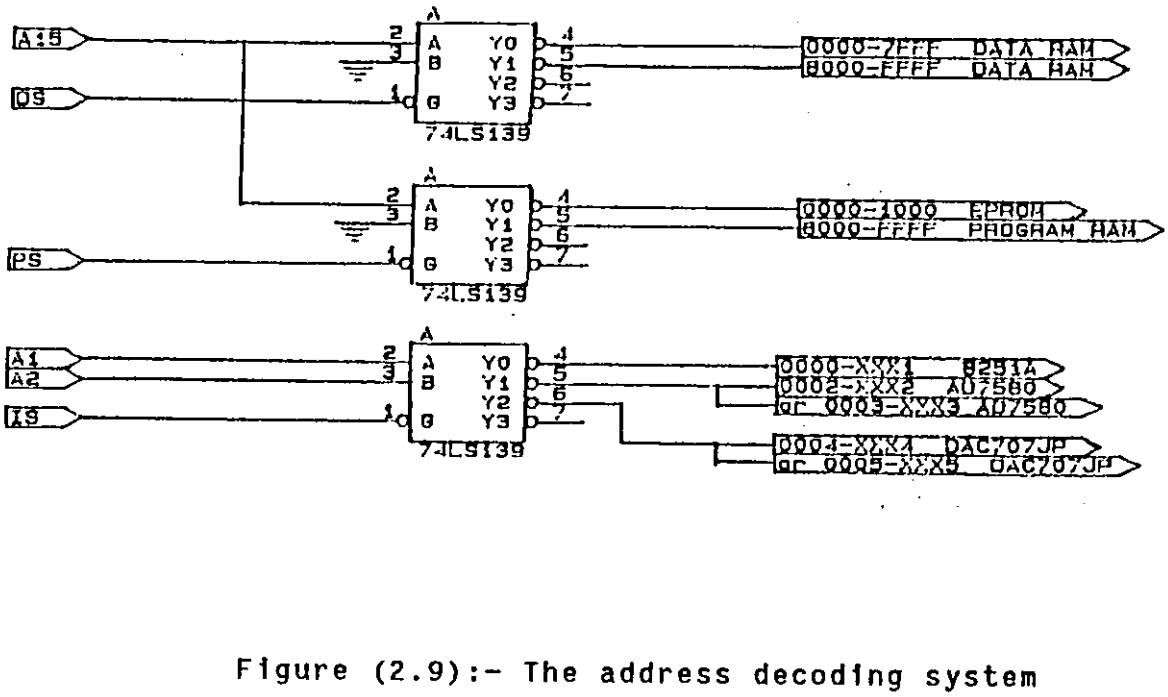


Figure (2.9):- The address decoding system

2.7 MISCELLANEOUS

1. RESET Signal of the board:-

The system contains a reset circuit, which is used to reset the DSP system by pressing a push button for a few seconds. The following ICs are affected by this reset operation:

1- TMS32020 microprocessor

2- 8251 A UART

3- DAC707 JP D/A converter

4- Flipflops that control the interrupts

2.CRYSTAL OSCILLATOR.The TMS32020 is clocked internally, using an external crystal of 20.00 MHz connected between pins X1 & X2/CLKIN of the processor.

3. POWER SUPPLY of the board

As mentioned previously, the board requires a dual power supply with voltage $\simeq \mp$ 15 Volts and $\simeq$ 3A output. These voltages are dropped and regulated to $\mp$ 12V. A +5V is regulated from the + 15 Volt to supply most of IC's on the board. The load is calculated theotrically to be 2A maximum. Practically with NO-load operation, the load reaches around 0.7A, the No-Load means just to power ON the board without programming or functioning.

CHAPTER 3

SOFTWARE

3.1 INTRODUCTION:-

In this chapter the operation of the DSP system is discussed. The various programs that control the operation and communication between the personal computer and the DSP board are also illustrated. The personal computer is connected to the DSP board by an RS232 cable through the serial port. The PC requires a program capable to communicate via the RS232 serial port. As an example, Kermit program is used here to transmit & receive data between PC & DSP system.

The main menu of the DSP system is shown in Fig (3.1).This menu appears on the screen if the system is initialized and the user send CR (Carriage Return) to the board after power up. The flow chart of the main program is shown in Figure (3.2).

Main Menu of the Program (Help menu)

```
        Welcome to Digital Signal Processing System DSP64
                    Using TMS32020 Processor
                      Done By Nidal Dali



PRESS:-


H              For Help (goback to main menu)
L (aaaa,llll)  For Loading a program into the DSP Board.
R (aaaa)       For Running a Program From the DSP Board.
D (aaaa,llll)  For Displaying Contents of Memory. (Data memory only)
F (aaaa,llll)  For Filling the Memory. (Data memory only)
    Your Choice:-
```

Fig. 3.1

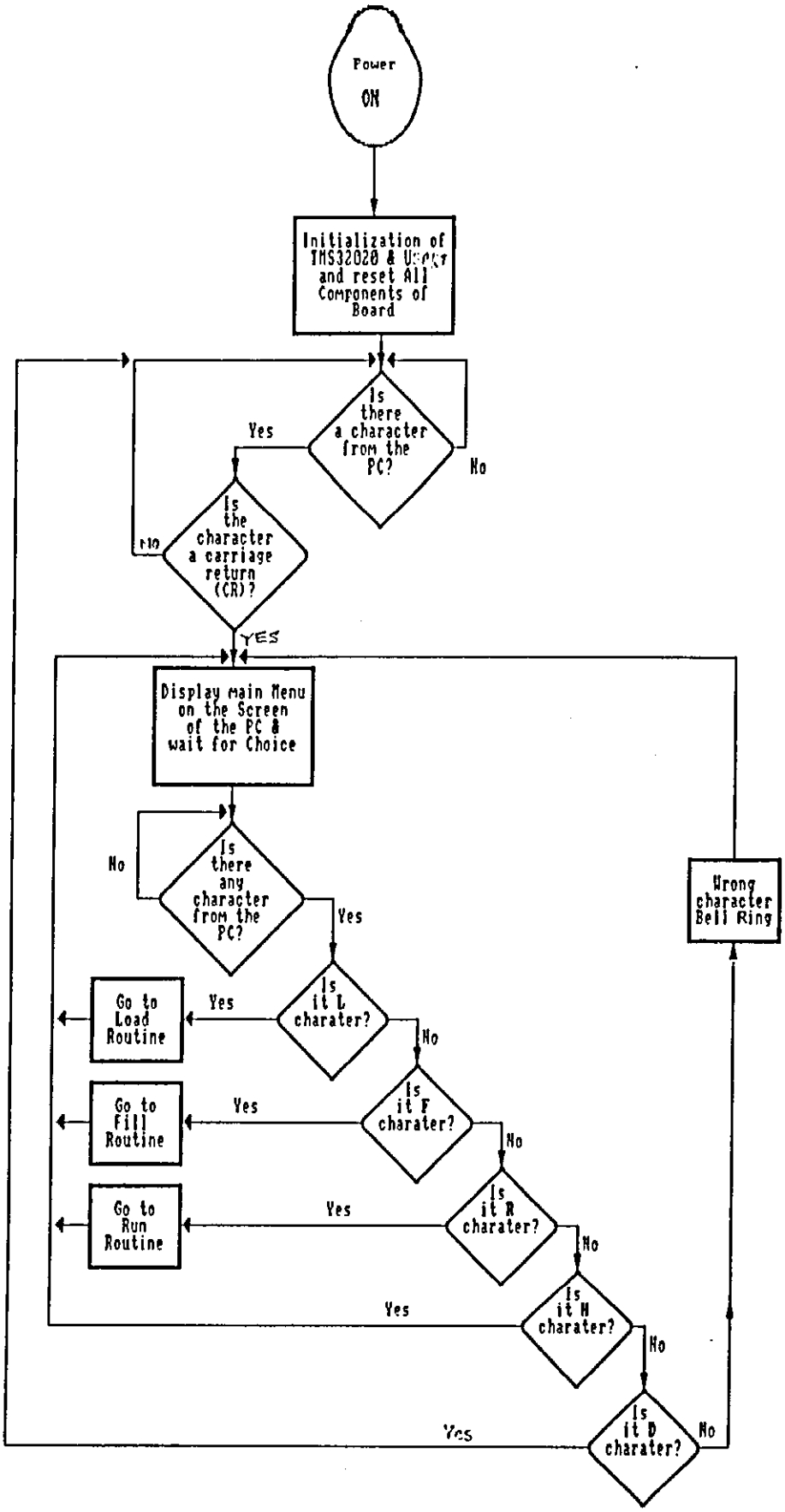


Figure 3.2 Main Programs

3.2 INITIALIZATION;-

After powering up the DSP system, an internal program is activated to initialize the microprocessor and the communication port. This program is also activated whenever a hardware reset is applied to the board by pushing the reset button.

The initialization program for the microprocessor is shown by the flowchart of fig.(3.3). The initialization programfor the USART is shown in the flowchart of fig. (3.4).

After initializing the TMS32020 & the 8251A, the DSP system waits for a Carriage Return (CR) from the PC. Then the main menu appears on the screen, and the DSP system waits to receive a character.

The main menu and its choices are discussed in the next sections.

3.3 MAIN MENU & PROGRAMS

The main menu of the DSP64 system is shown in fig.(3 .1). The user can choose any of the five choices in a specific format.

The notation "aaaa" represents a 16-bit address in HEX format, and the notation "llll" represents a 16-bit length in HEX format. The use of these notations depends on the command chosen.

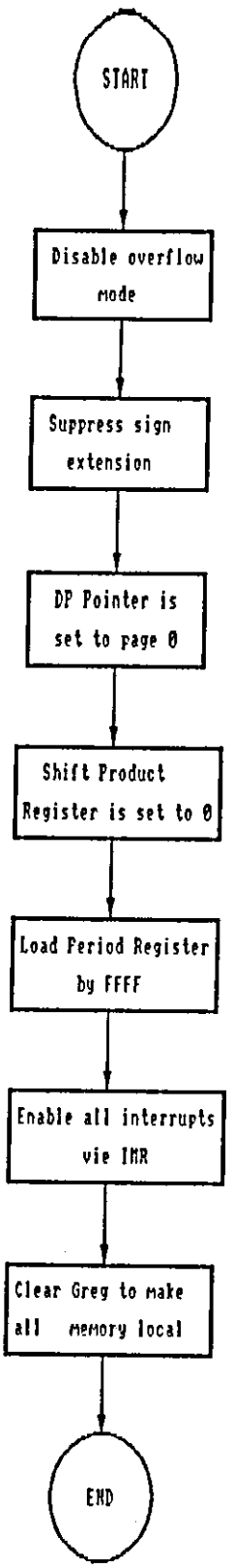


fig.(3.3):- The initialization program for the microprocessor

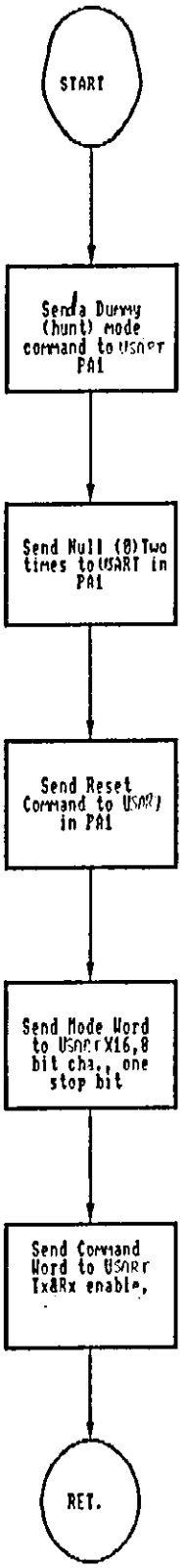


Figure 3.4

Initialization of the USART8251A

The user may use any of the offered choices. For example, he can initiate the DSP screen by pressing H for help and redisplay the main menu. The user may load a program into the DSP64 or run a program starting at a specific location. The user may display the data in the memory, or fill this data for special applicaiton.

An explanation for each command are discussed in the following sections:-

3.3.1 PRESSING H will redisplay the main menu and wait for a choice. Displaying the main menu is shown in the flowchart of fig.(3.5) and processing the user choice program flowchart is shown in Fig.(3.6).

3.3.2 PRESSING L (aaaa,llll) causes the DSP main program to go to the loading program and prepares the DSP64 system for receiving the program and saving it at location (aaaa) in the program memory space with a length of (llll). The flow chart of loading program is shown in Fig.(3.7).

The load program at first prepares the address and the length of the program to be loaded and when ready it tells the user by displaying the message "Load Data" on the screen.

The Load-program subroutine receives the comming data and

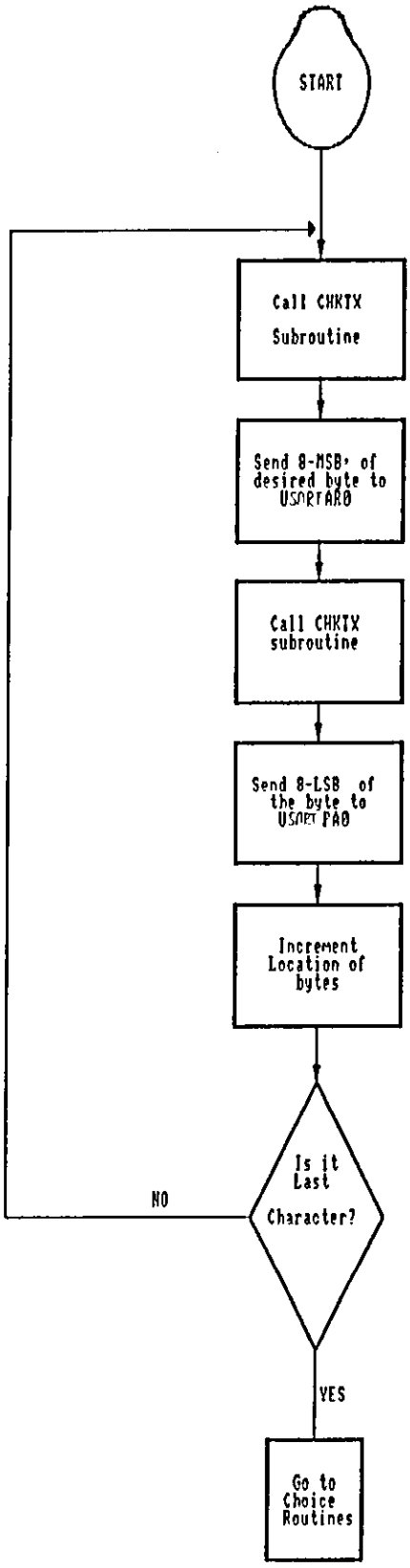


fig. (3.5)

(MAIN MENU)
Display main menu routine

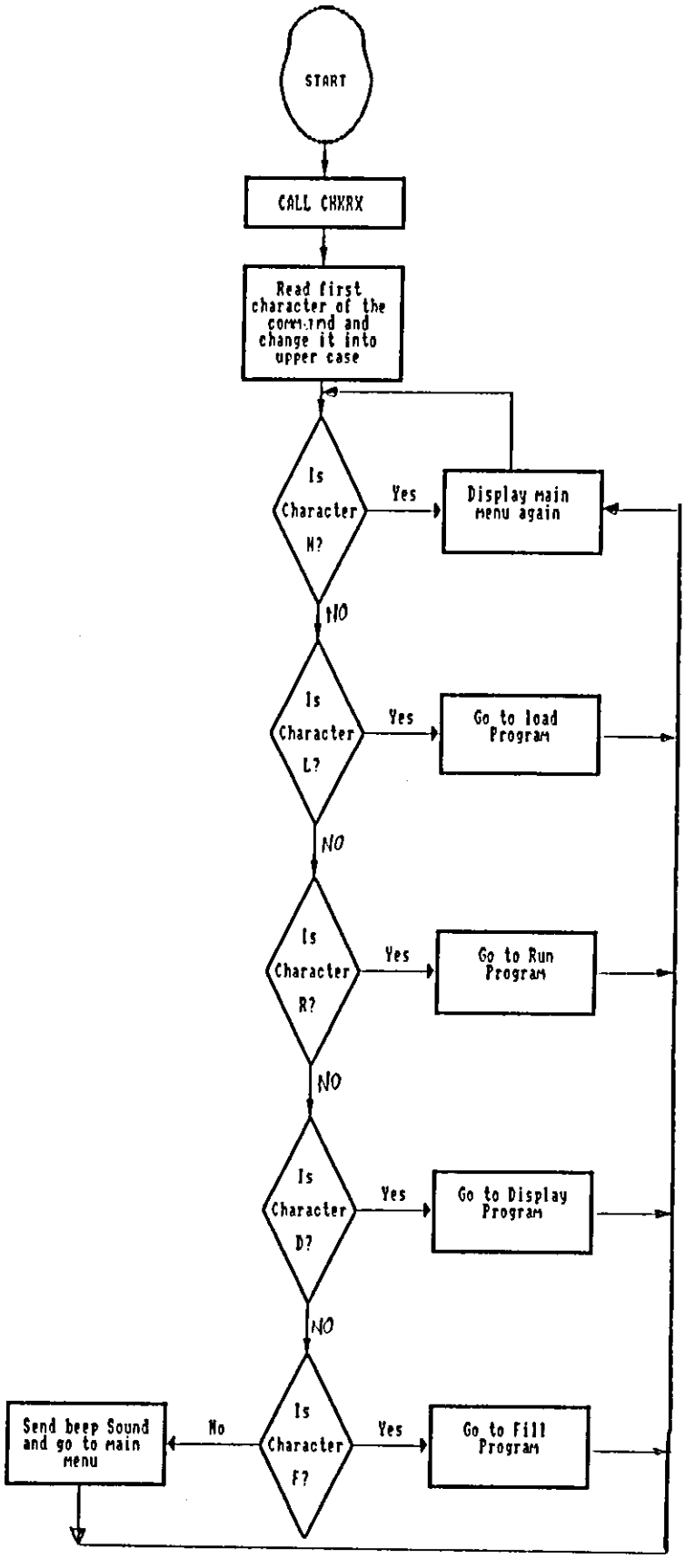


Figure 2.6 Choice Program

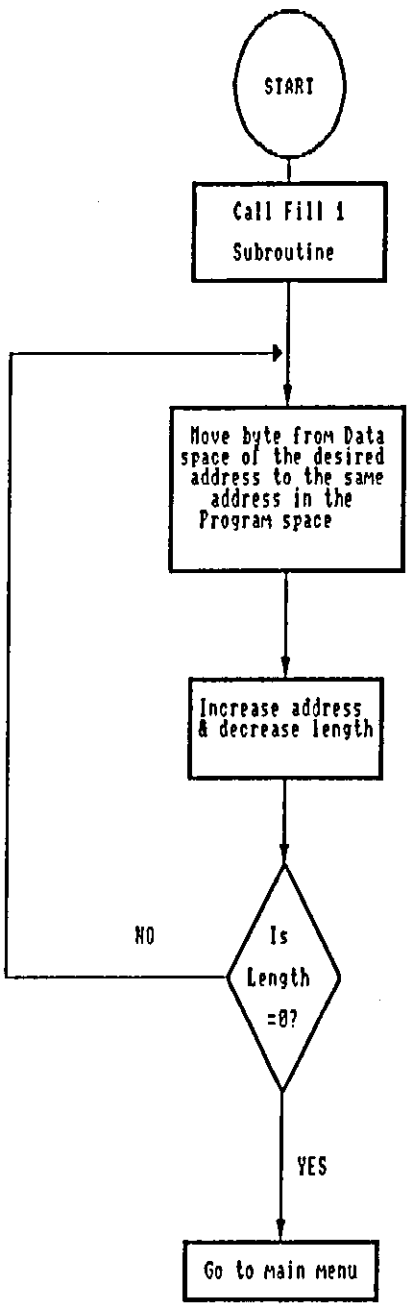


Figure 3-7 Load Program

saves it in the data memory space. Then it moves this data into the program memory space. So the locations in both data and program memory spaces should be empty or the data in it will be lost. After loading the user program, the main DSP program goes to the main menu.

3.3.3 PRESSING R (aaaa) will run the program stored in the program memory at location (aaaa). After execution, the main program returns to the main menu.

The flowchart of the RUN program is shown in Fig. (3.8)

3.3.4 PRESSING D (aaaa,llll) will Display on the screen the data stored in the data memory at address (aaaa) and length (llll).

After displaying the data, the main program waits for a Carriage Return (CR) to go to main menu. The flowchart of the Display program is shown in Fig. (3.9).

3.3.5 PRESSING F (aaaa,llll) will make the DSP main program go to the filling program and prepare the system for receiving the data to be filled starting at location (aaaa) and length (llll) in the data memory. A message is sent to the PC screen "Load Data" to tell the user that the DSP64 system is ready. After filling the data, the main program goes to the

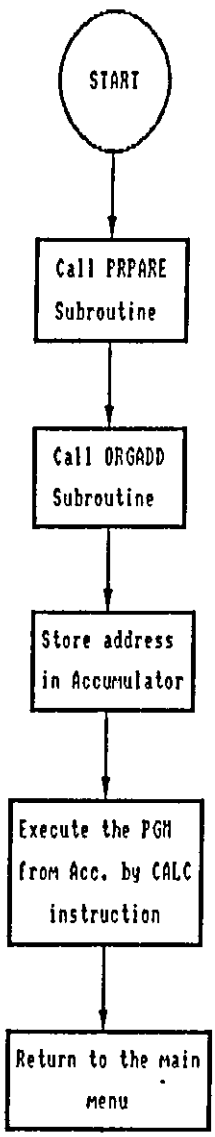


Figure 3.8 RUN PROGRAM

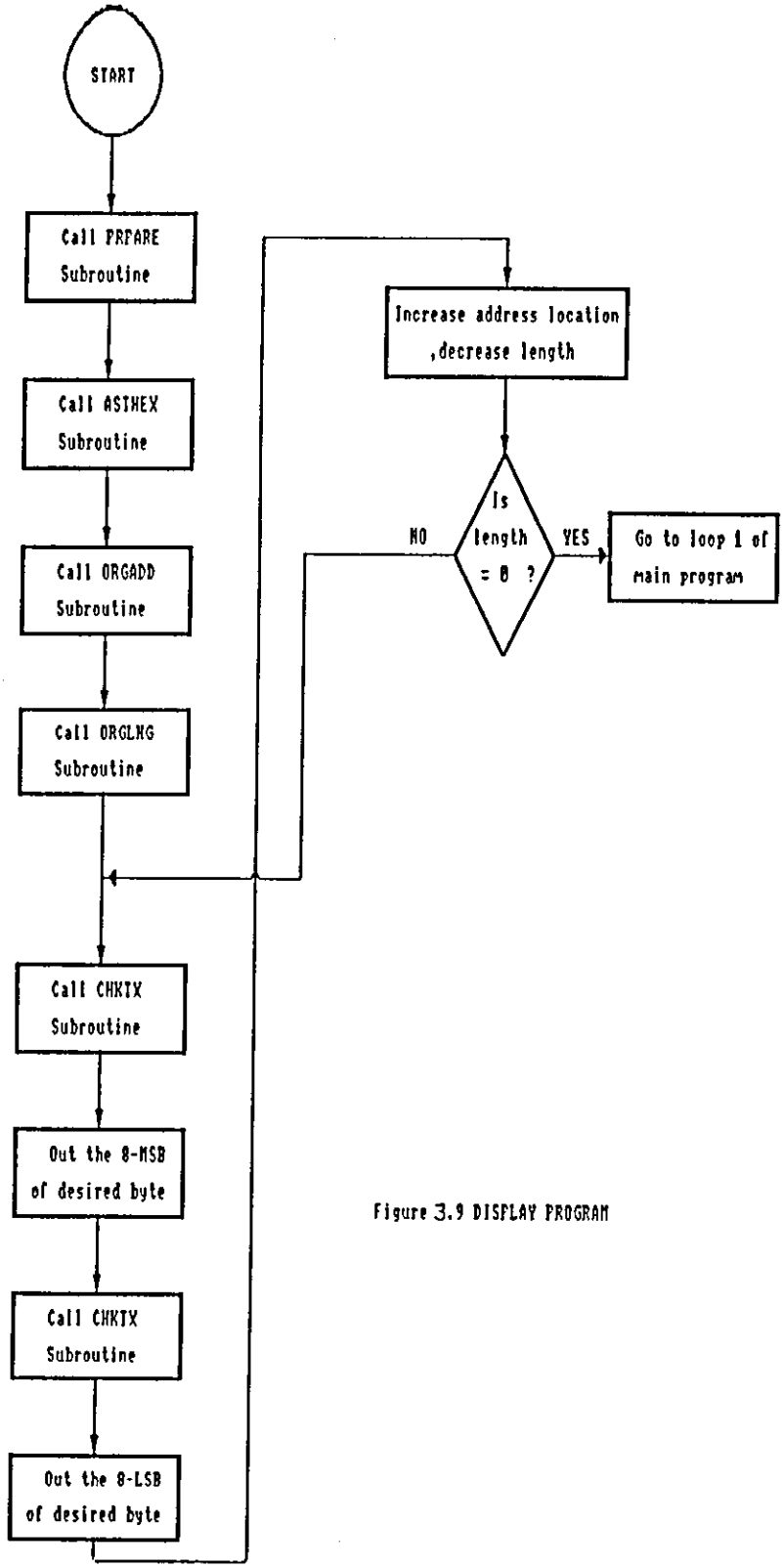


Figure 3.9 DISPLAY PROGRAM

main menu. This command is like the load command except that the data is not moved to program space. The flowchart ofFill program is shown in Fig. (3.10).

3.4 Subroutines:-

In this section, different subroutines that are used in the main programs will be discussed and a flowchart for each subroutine will be presented.

3.4.1 CHKRX SUBROUTINE

This subroutine is used to check if the USART is ready to Receive data from the PC or not. This program reads and checks the status register over and over until it finds D1; the RxRDY bit high. When it finds that the receiver is ready with a character, it reads the status register again to find out whether any errors were detected. If any error was detected, the programs goes to the main menu. A flowchart of CHKRX subroutine is shown in Fig.(3.11).

3.4.2 CHKTX SUBROUTINE:

This subroutine is used to check if the USART is ready to transmitt data to the PC or not. This program reads and checks the status register over and over until it finds that DSR (Data SET Ready) signal is activated and the transmiter is ready (TXRDY). A flow chart of CHKTX subroutine is shown in Figure (3.12).

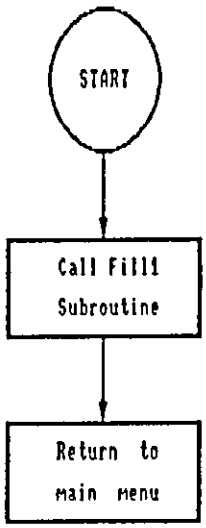


Figure 3.10 FILL PROGRAM

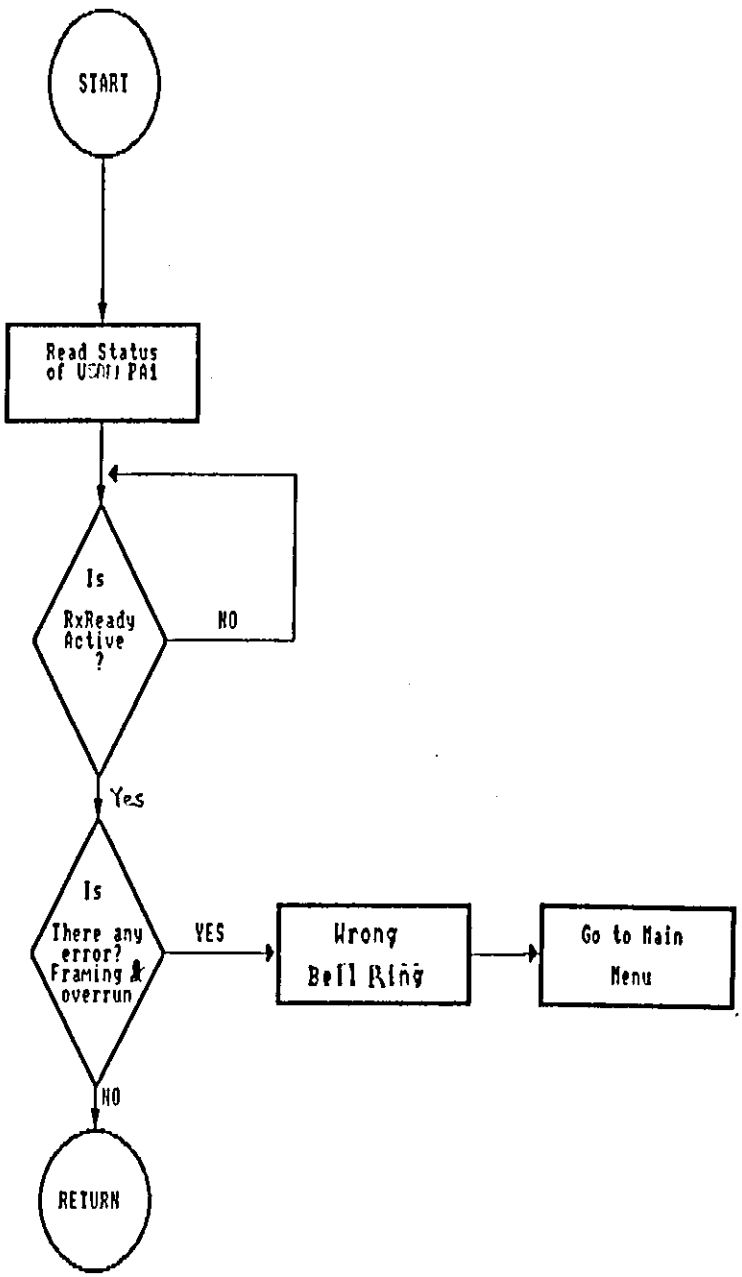


Figure 3.11 CHKRX Subroutine

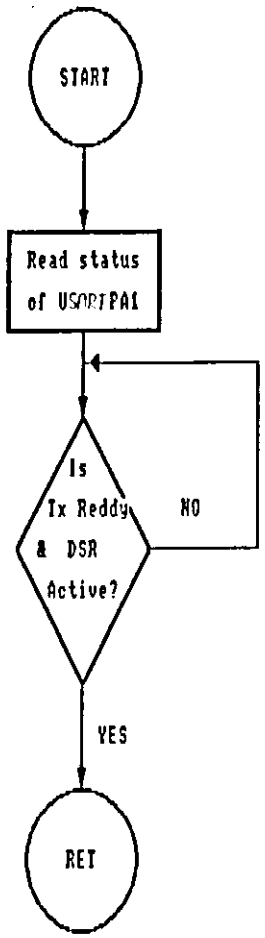


Figure 3.12 CHKTX Subroutine

3.4.3 PREPARE SUBROUTINE:

Prepare subroutine is used to receive the command format (aaaa,llll) or (aaaa) which is the address and the length of the parameters of the command. It stores the address in memory locations 0061H - 0064H and the length in 0066H - 0069H as follows:

>0060	ASCII of (
>0060	" " a MSBa
>0062	" " a
>0063	" " a
>0064	" " a LSBa
>0065	" " ,
>0066	" " l MSBl
>0067	" " l
>0068	" " l
>0069	" " l LSBl
>0070	" ")

In this subroutine, Auxiliary Register AR0 is used to point to location >0061 for indirect addressing, Fig.(3.13). shows the flowchart of PREPARE subroutine.

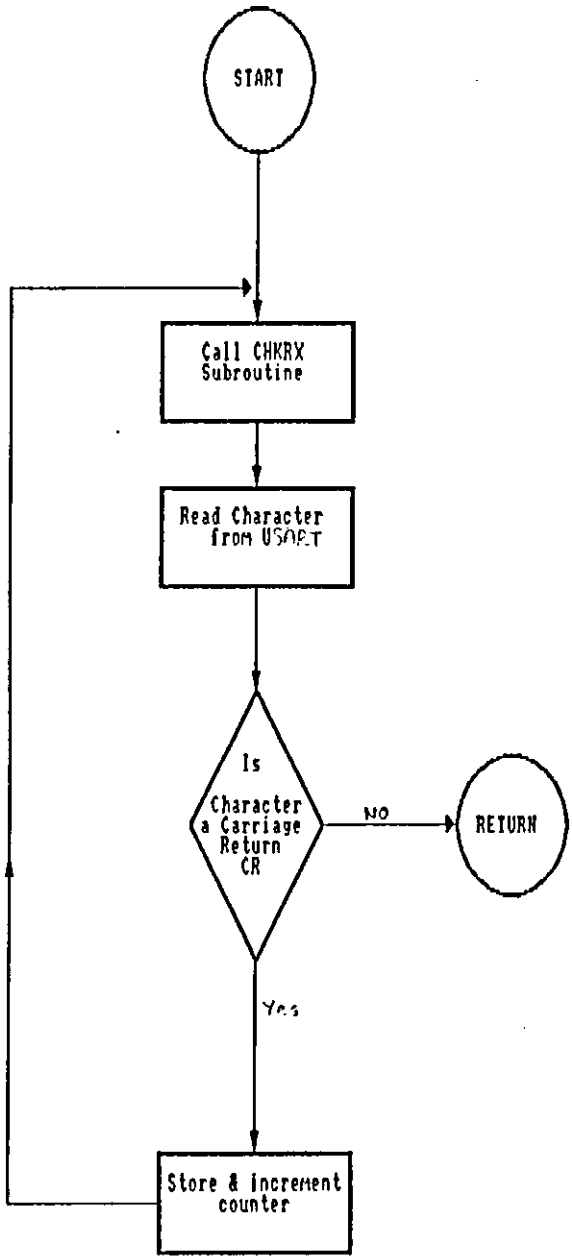


Figure 3.13 Prepare Subroutine

3.4.4 ORGADD SUBROUTINE:

This subroutine is used to assemble the (address) that is stored in location 0061-0064 in Hex format as shown in the table aside. (a MSB) is the 4-MSBs of the address and (a LSB) is the 4-LSBs of the address.

0061	a MSB
0062	a
0063	a
0064	a LSB
0065	am a a a1

The assembled address in the form of a 16 - bit number is stored in location 0065 H of the data memory.

The flowchart of the ORGADD subroutine is shown in Fig.(3.14).

3.4.5 ORGLNG SUBROUTINE:

This subroutine is used to assemble the (length) that is stored in locations 0066-0069 in Hex format as shown in the table below:-

0066	l MSB
0067	l
0068	l
0069	l LSB
0070	l l l l

l MSB is the 4-MSBs of the length,
l LSB is the 4-LSBs of the length.

The assembled length in the form of a 16 - bit number is stored in location 0070 of the data memory. The flowchart of the ORGLNG subroutine is shown in Fig.(3.15).

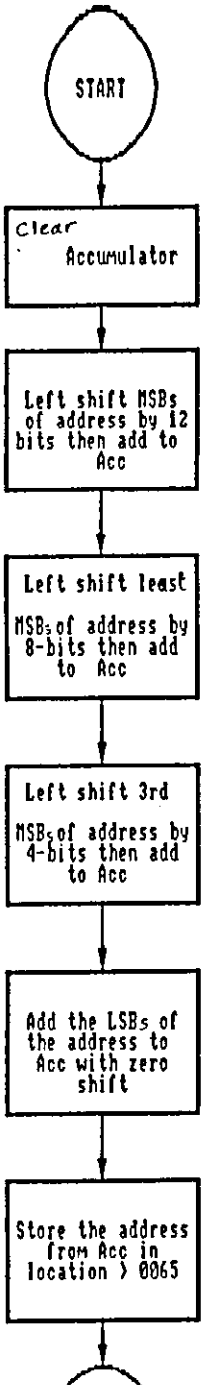


Figure 3.14 Orgadd Subroutine

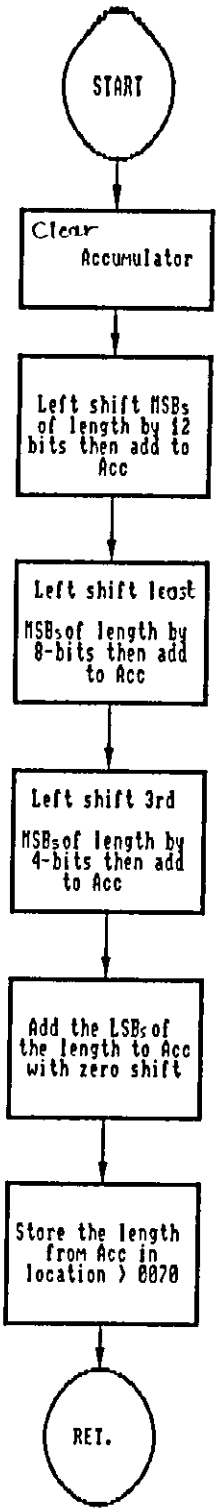


Fig. 3.15 ORGLNG

3.4.6 ASTHEX SUBROUTINE:

This subroutine is used when a character is received by the system in the ASCII code. this subroutine converts the ASCII character into Hex (Binary) representation to enable the processor to process the character. This subroutine is applied only for the hexadecimal characters 0-F Auxiliary register AR0 points to the first character and Auxiliary register AR1 points to number of characters. Each character is changed into Hex format and stored in its same location. The flowchart of the ASTHEX subroutine is shown in Fig.(3.16).

3.4.7 FILL1 SUBROUTINE

This subroutine is used for loading & fillingprograms from the PC. It is used to fill a stream of data starting in locations in the data memory addressed by Auxiliary register AR3. The flowchart of the fill1 subroutine is shown in Fig.(3.17).

The assembly programs and subroutines in the language of the TMS32020 are presented in the appendix at the end of this thesis. In the next chapter, an applications and sample programs will be shown and discussed.

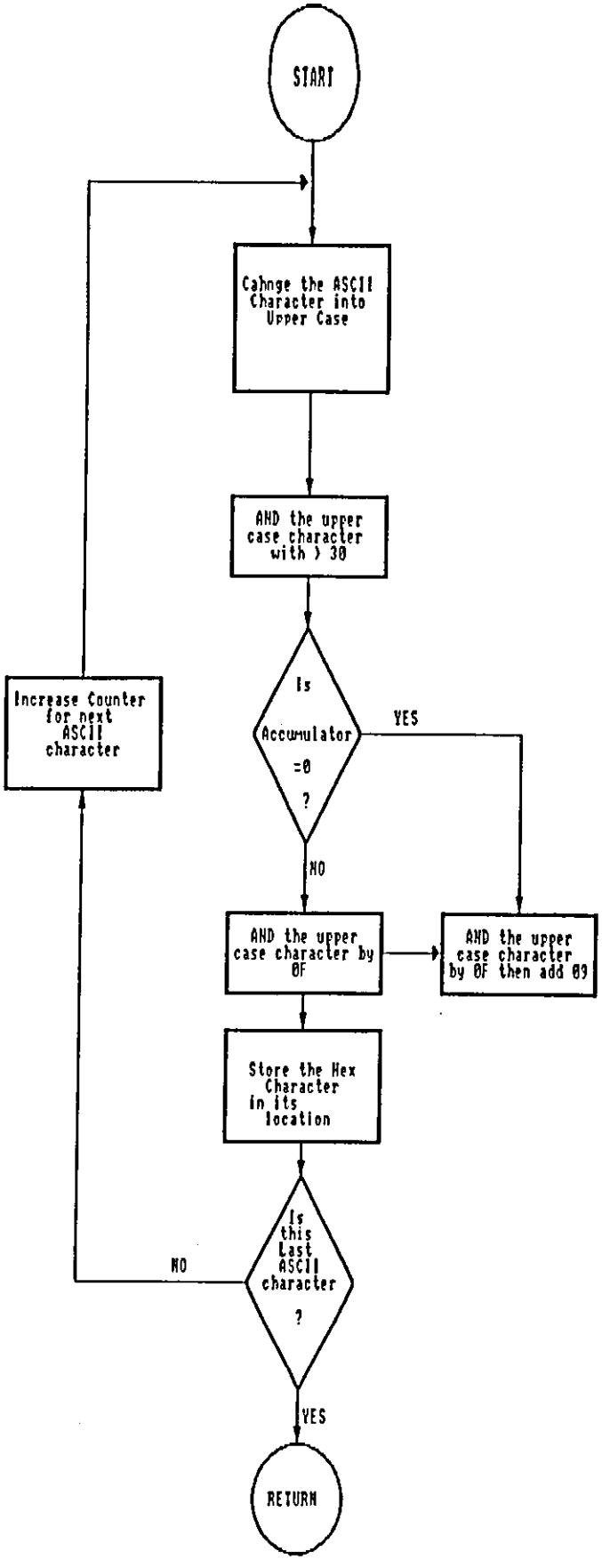


Figure 3.16 ASTHEX Subroutine

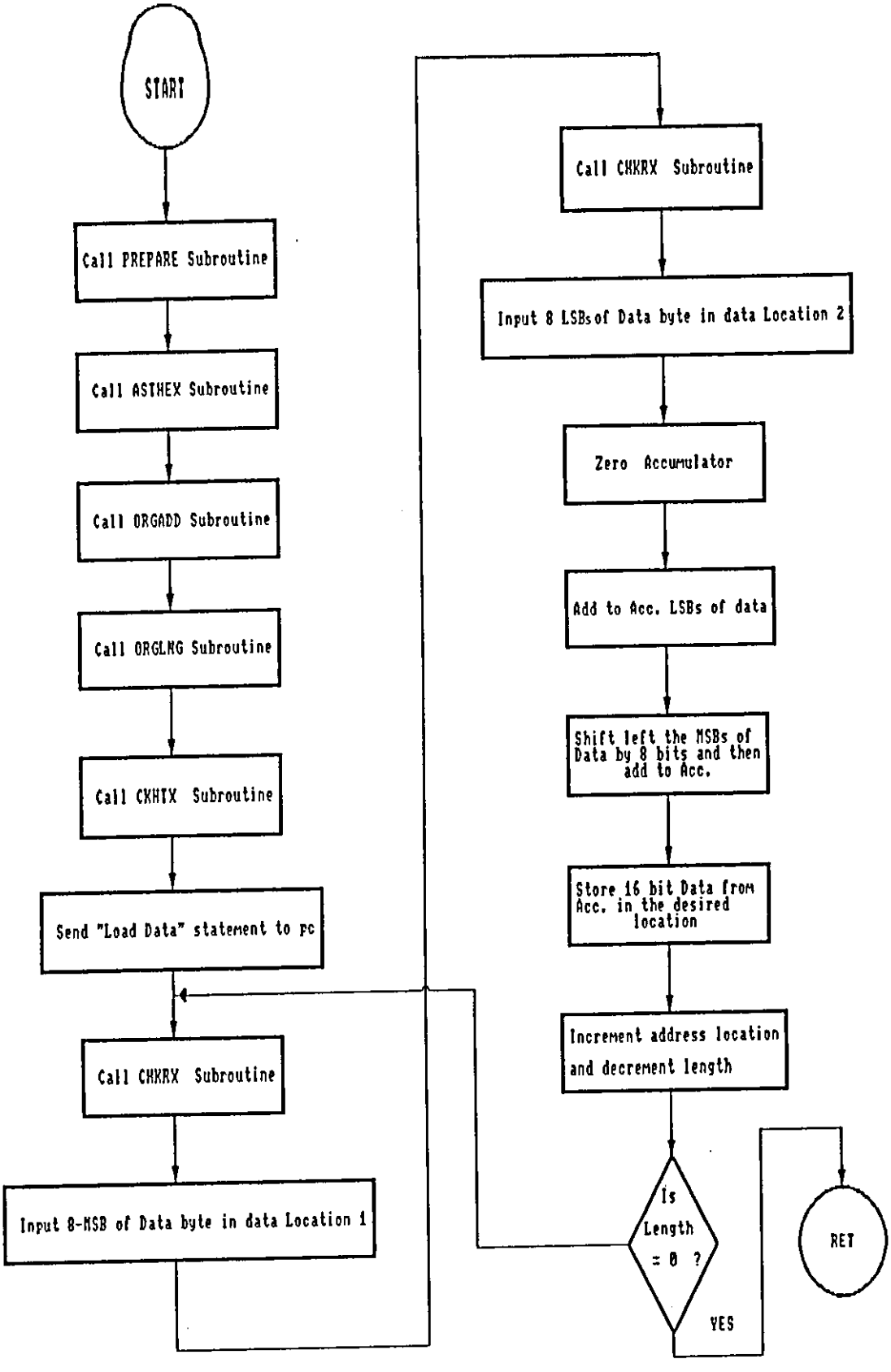


Figure 3.17 Fill 1 Subroutine

CHAPTER 4

APPLICATIONS

The digital signal processing board DSP can be used in many applications, it may be considered as a general purpose digital signal processing system.

It has an analog input channel and, an analog output channel. A reasonable data and program memory in a portable system that can be used with any PC has a serial port. A variety of programs can be loaded into the board from the PC according to the required application.

Typical applications include[10]:-

* Digital filtering
* Correlation
* Windowing
* Fast Fourier transforms
* Adaptive filtering
* Waveform generation
* Speech analysis
* Speech synthesis
* Speech recognition

An example of one of these applications is the Implementation of FIR/IIR filters.

In many digital signal processing applications, it is

advantageous to use digital filters in place of analog filters. Digital filters can meet light specifications on magnitude and phase characteristics and eliminate voltage drift, temperature drift and noise problems associated with analog filter components. This application describes one method for implementing Finite Inputs Response (FIR) and Infinite Inputs Response (IIR) with the DSP system.

4.1 Advantages of Digital Filtering[15]:-

The term "digital filter" refers to a computational process or algorithm by which a digital signal or sequence of numbers representing an input is transformed into a second sequence of numbers termed the output digital signal. Digital filters involve signals in the digital domain (discrete-time signals), whereas analog filters relate signals in the analog domain (continuous - time signals).

A band limited continuous - time signal can be converted into a discrete-time signal by means of sampling. This is performed in the A/D part of the board by the analog input channel.

After processing, the discrete-time signal can be converted back to a continuous-time signal. Tthis is achieved by the Analog output channel and D/A converter.

Some of the advantages of using digital filters over their analog counter parts are:-

1. High reliability
2. High accuracy
3. No effect of component drift on system performance.
4. Component tolerances not critical.

Another important advantage of digital filters when implemented with the programmable processor TMS32020 is the ease of changing filter parameters to modify the filter characteristics. This feature allows the design engineer to easily upgrade or update the characteristics of the designed filter due to changes in the application environment.

4.1 DIGITAL FILTER IMPLEMENTATION[16]

For a large variety of applications, digital filters are usually based on the following relationship between the filter input sequence X(n) and the filter output sequence Y(n)

$$Y(n) = \sum_{k=0}^{N} a_k \, y(n-k) + \sum_{k=0}^{M} b_k \, x\,(n-k) \qquad (1)$$

equation (1) is refered to as a linear constant coefficient difference equation. Two classes of filters can be represented by linear constant - coefficient difference equations:-

1. Finite Impulse Response FIR filters.

2. Infinite Imnpulse Response IIR filters.

4.2.1. FIR FILTERS:[16]

For FIR filter, all of the a_k in (1) are zero therefore, (1) reduces to:

$$Y(n) = \sum_{k=0}^{M} b_k \; x(n-k) \qquad (2)$$

where (M+1) is the length of the filter. As a result, the output of the FIR filter is a finite length weighted sum of the present and previous inputs to the filter.

If the unit-sample response of the filter is H(n) then:-

$$Y(n) = \sum_{=0}^{M} h(k) \; x(n-k) \qquad (3)$$

where h(n) = B(n), taking the Z - transform

$$H(z) = \frac{Y(Z)}{X(Z)} = \sum_{k=0}^{M} b_k \; Z^{-1} = \sum_{k=0}^{M} h(k) Z^{-k} \qquad (4)$$

equations (3) & (4) can also be represented by the network structure shown in figure 3.1

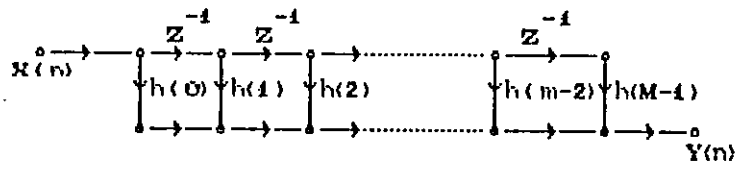


Figure 4.1 :- Network structure for FIR filter (1).

This structure represents a direct-form realization of an FIR filter. The branches of fig. 4.1 labeled as Z^{-1} corresponds to the delay in (3) and the multiplications by Z^{-1} in (4).

4.2.2 TMS32020 IMPLEMENTATION OF FIR FILTERS[10]

The notation XN used here corresponds to x(n), and XNM1 corresponds to x(n-1).

In the above implementation, the following three basic and important concepts for the implementation of FIR filters on the TMS32020 should be understood:

1. The relationship between the unit sample response of an FIR filter and the filter structure.
2. The power of LTD, MPY, RPTK, MACD instructions for this implementation.
3. The ordering of the input samples in the data memory of the TMS32020.

The input sample sequence x(n) and the unit sample response h(n) are stored as shown in figure 4.2

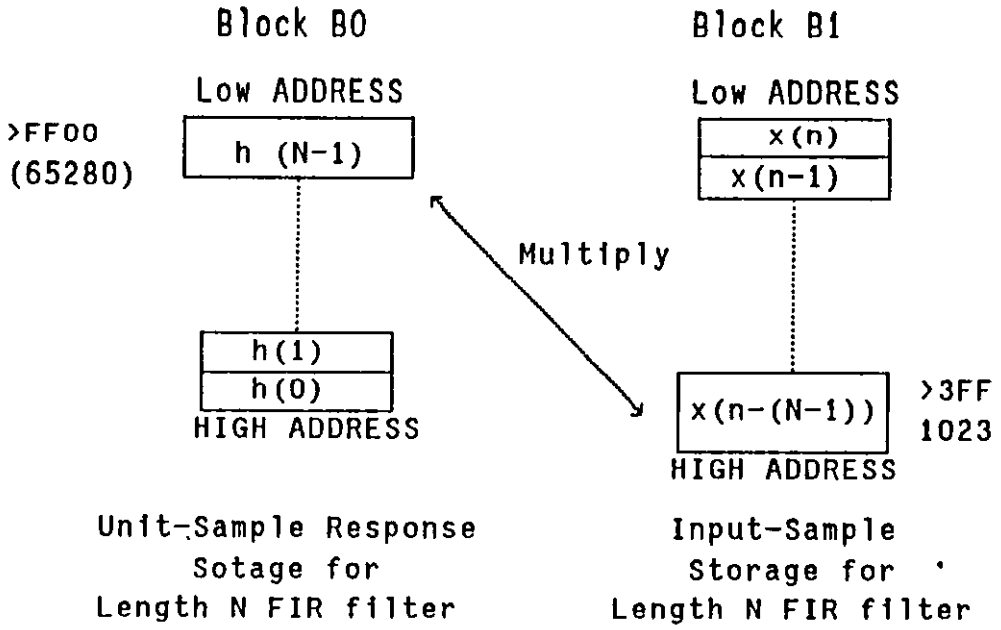


Figure 4.2:- Storing input sample response (1).

In general, each of the multiplies and shifts of x(n) in equation (3) is implemented with the instruction MACD (multiply and accumulate with data move). Block B0 must be configured as program memory using the CMFP instruction. MACD only works with on-chip RAM. The use of MACD instruction helps to speed the filter execution and allows the size of the FIR filter to expand to 256 taps. The TMS32020 implementation of (3) is made even more efficient with a repeat instruction, RPTK. It forms a useful instruction pair with MACD such as

```
RPTK    NM1
MACD    (PMA), (DMA)
```

The RPTK NM1 instruction loads an immediate 8-bit value N-1 into the repeat counter, this causes the next instruction

to be executed N times (N = the length of the filter). The Instruction MACD (PMA),(DMA) performs the following functions:-

1. Loads the program counter with PMA
2. Multiplies the value in data memory location DMA (on-chip block B1) by the value in program memory location PMA (on-chip block B0)
3. Adds the previous product to the accumulator.
4. Copies the data memory value (Block B0) to the next whereh(n)=B(n)higher on-chip RAM location. The Data move is the mechanism by which the Z^{-1} delay can be implemented.
5. Increments the program counter with each multiply/ accumulate to point to the next sample of the unit-sample response.

Figure 3.3 shows a code example by the TMS32020. Data memory values are accessed indirectly through auxiliary register AR1 when the MACD instruction is implemented.

For low-order filters (second-order) using the MACD instruction in conjunction with the RPTK instruction is less effective due to the overhead associated with the MACD instruction in setting up the repeat construct. To take advantage of the MACD instruction, the filter must be greater

than three. For lower order filters, it is better to use LTD/MPY instruction pair in place of RPT/MACD.

The TMS32020 provides a solution for faster execution of FIR filters. The combination of the RPTK/MACD instructions provides a minimum program memory and high speed execution of the FIR filter. If data memory is of concern, the designer can use LTD/MPYK instruction pair. Figure 4.3 shows the TMS32020 code for implementation of FIR filter.

```
*     This section of code implement the equation:-
*     X(n-(n-1) h(N-1) +X(n-(N-2) + ... + X(n) h(0) = Y(n)
        CNFP              * USE BLOCK B0 AS PROGRAM AREA.
NEXRPT  IN     XN,PA2     * BRING IN THE NEW SAMPLE XN FROM A/D
        LRLK   AR1>3FF
        NPYK   0          * SET P REGISTER TO ZERO.
        ZAC               * CLEAR ACCUMULATOR
        RPTK   NM1        * REPEAT N-1 TIMES.
        MACD   >FF00      * NULTIPLY / ACCUMULATE
        APAC
        SACH   YN,1
        OUT    YN,PA4     * OUTPUT THE FILTER RESPONSE BY D/A
        B      NEXTPT     * GET THE NEXT POINT.
```

Figure 4.3:- TMS32020 code for implementation of FIR filter

4.3 IIR FILTERS[10]

For IIR filters, at least one of the a_k in equation (1) is nonzero. The Z transform of the unit-sample response of an IIR filter corresponding to (1) is

$$H(Z) = \frac{Y(Z)}{X(Z)} = \frac{\sum_{k=0}^{M} b_k Z^{-k}}{1 - \sum_{k=1}^{N} a_k Z^{-k}} \qquad (5)$$

Three different network structures are often used to implement (5): the Direct form, the Cascaded form, and the Parallel form. Implementation of Direct-Form IIR filter is discussed next.

Equations (1) & (5) can also be represented by the network structure shown in figure 4.4

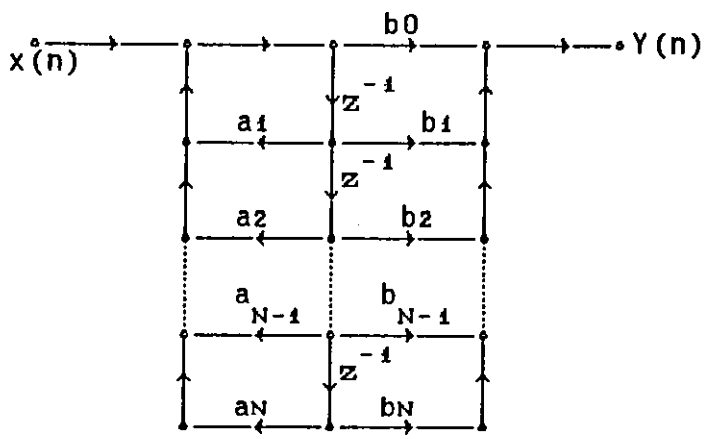


Figure 3.4:- Network structure for IIR filter (1).

For convenience, it is assumed that M = N.

This network structure is referred to as the

direct-formII realization of on N^{th}-order difference equation. The branches associated with the Z^{-1} correspond to the delays in (1) and the multiplications in (5).

The following difference equation:-

$$Y(n) = \sum_{k=1}^{N} a_k\, Y(n-k) + \sum_{k=0}^{M} b_k\, x(n-k) \qquad (6)$$

Shows that the output of the filter is a weighted sum of past values of the input to the filter and of the output of the filter.

The difference equation for figure 4.4 is

$$d(n) = x(n) + a_1 d(n-1) + a_2 d(n-2) + \ldots + a_N d(n-N)$$

$$Y(n) = b_0 d(n) + b_1 d(n-1) + b_2 d(n-2) + \ldots + {}_N a\, d(n-N)$$

where d(n) corresponds to the network value at different delay nodes as shown in figure 4.5

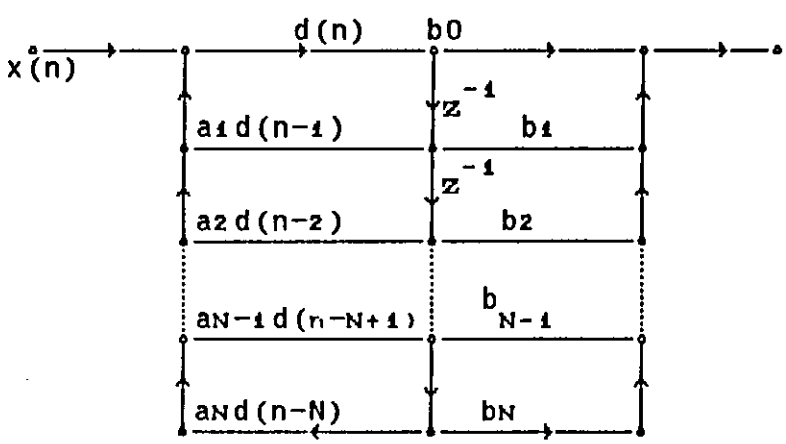


Figure 4.5:- Another Network structure for IIR filter (1)

The zero-delay register corresponds to d(n); d(n-1) is the register for delay of one; the d(n-2) is the register for the delay of two.

The delay-node values of the filter are stored in data memory as shown in figure 4.6.

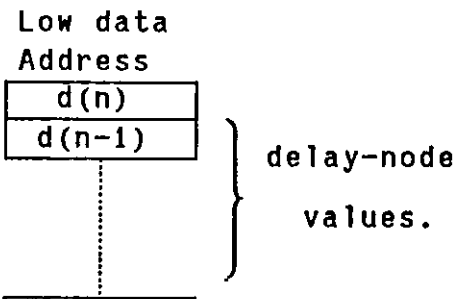


Figure 4.6:- Delay node values stored in data memory.

Figure 3.6 shows the necessary ordering of the delay-node values for a general direct-form II structure for the case M≥N. Filter order is determined by M or N, whichever is a greater.

Figure 4.7 shows a portion of the TMS32020 code for implementing the direct-form II IIR filter.

```
NEXT    IN      XN,PA2    *    Input new value XN
        LAC     XN
        MPYK    0         *    Clear P register
        LARP    AR1
```

```
        LRLK    AR1,>03FF
        CNFP                    *   Use block B0 as program area
* d(n) = x(n) + d(n-1) a1 + d(n-2) a2 + ... +d(n-N) aN
        RPTK    N-1             *   Repeat N times
        MACD    >FF00,*-
        APAC
        SACH    DN,1            *   d(n)
* Y(n) = d(n) bo + d(n-1)b1 + d(n-2) b2+ ...+ d(n-M) bM
        ZAC
        MPYK    0
        MPY     >(FF00+N)
        RPTK    N - 1
        MACD    > (FF00 + N+1), *-
        APAC
        SACH    YN,1            *   Saved filtered output
        OUT     YN,PA4
        B       NEXT
```

Figure 4.7:- TMS32020 code implementation for FIR filter.

Finally, these are two examples of implementing digital filters type FIR & IIR with the digital signal processor TMS32020 that can be applied on the DSP 64 board. There are many examples that can the user do like waveform generation fast fourier transforms and speech analysis & synthesis.

CHAPTER 5

RESULTS AND DISCUSSIONS

In this chapter the results of troubleshooting, operation and testing of the DSP board will be presented and discussed. The printed circuit board of the project was desinged by using a computer program, taking into consideration minimization the size of the board to reduce the length of the tracks between parts. The board was designed as a double sided board and printed in the electrical enigeering laboratories. Afrer printing , the troubleshooting starts by checking all tracks and component places, some errors appeared and fixed, then all sockets of ICs and other components were welded, also another cheking is made without power. Then, and without the components, the board is tested by putting the power supply on, all areas were tested to have the power . The microprocessor is installed in it's location and given the condition of No-operation instruction externally to the data bus, then by using the oscilloscope, all control signals, data bus, and address bus were checked.

In this type of boards, the oscilloscope is not enough to monitor the signals because it is changing at high speed (5 million instruction per second), so, a logic analyzer is used to monitor the data

bus, address bus, read/write singals and chip selects for most components. In the No-operation Condition, the address is increased from 0000 to FFFF and all other control signals were captured by the logic analyzer and checked to be true.

The next stage is checking that all ICs can be selected by the micro processor, a small program was programmed the EPROMS of the program space of the microprocessor to select each IC related to the TMS32020, the result was that each IC is selected according to the program correctly and by monitoring that on the logic analyzer. At this stage, it seems that the processor is working correctly and all control singals out from the processor were in right timing and correct direction.

The next step is to operate each zone of the board by a program, the first zone which tested was the RS232 serial communication port, several programs were tested to operate this serial port (shown in the Appendix [17]), different programs were tested and a lot of time is spent to ensure the communication correctly.

Actually, the job of the serial communication port is to

download the programs from the PC to the program area in the board, and because of the fail to operate the serial port always correctly, the trend went to download the programs directly by an EPROMS to the program area of the board, and in this way, one can operate the board externaly without using the PC and on the other hand, the programs will be always kept in the board even if the power is off, the board will be operated with one function until another program is installed in the EPROMS.

In the next stage, testing of the A/D to Digital area is done by inserting control signals to the AD7580 IC directly and externaly on the DSP board, and by applying different input voltages to the analog input channel, the data bus is read by the logic analyzer and the digital value approximatly corresponds to the analog input signal, a program is written on the EPROM to operate the A/D zone of the board by the CPU (shown in the appendix), different programs were used but it didn't work by the order of the microprocessor every time.

The same problem appears in the D/A converter zone of the board, program is written to generate square wave from the D/A converter , the DAC707 IC didn`t response to the signals from the

CPU always.

On the other hand it works when tested outside the board on a bread board. Several programs were tested, all possible troublshooting is applied on D/A, A/D zones of the board. The signals were followed up by the logic analyzer, everything run normally from the microprocessor side, but on the other side it didn't response always. Many tests were applied to the timing of the control signals out from the microprocessor, some components were changed with a higher speed one's, software delays for the control signals in the software programs were applied.

Up to this point the research with the capabilities available didn't reach to the solution of this proplem and may need some higher technology experiences in this field.

CHAPTER 6

CONCLUSIONS & RECOMMENDATIONS

This work covers a special microprocessor used for special applications, it works relatively with high speed execution time (5 million instruction per second), these types of microprocessors need a special way of designing the printed circuit board.

The circuit diagram of this board is done taking into consideration the recommendations published by Taxas Instruments (the source of TMS32020) and by all the design techniques available , all parts used in this board were special, expensive componets with high quality and high reliability and used in such designs, most of these part were imported specially for this project .

I tried in designing the Printed Circuit Board to make the design suitable for such sensitive components, and to put all possible solutions to avoid noise, ringing high peaks and other influences that may affect the operation of the board. The board was designed as a double sided board, the welding of IC sockets was so difficult because of manual welding, the quality of

welding is not excellent for such sensitive boards.

In this project, there was a noise effect and a poor quality of printed circuit board manufaturing, such board needs a high technology in PCB manufacturing, sheilds must be inserted over the important and sensitive ICs such as the TMS32020 and the AD7580 and DAC707.

I recommend in future desings to weld the ICs in the board directly without using IC sockets specially for the TMS32020, A/D and D/A channels, and insert special sheilds to protect the ICs, from outside influences.

Finaly; this design of the DSP system needs to be executed in a high technology expert center to avoid all disturbances that may occur from poor implementation on PCB.

REFERENCES

REFERENCES

1- Surendar S.Magar,Edward R.Caudel and Anthoy W.Leigh. " Session II, Digital Signal Processing. A microcomputer with Digital Signal Processing Capability," IEEE INTERNATIONAL SOLID STATE CIRCUITS CONFERENCE, February 1982, pp 80-86.

2- Kevin C. Mc Donough and Surendar S. Magar. " A single chip microcomputer Architecture optimized for signal processing ," IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING",May (3-5), 1982, pp 101-107.

3- Bruce Secrest ," Speech analysis and synthesis become practical on µC chip, "ELECTRONIC DESIGN, , May 27, 1982, pp 129-135.

4- Richard Pickvance, " A single chip digital signal processor , part 1: architecture and addresssing , "ELECTRONIC ENGENEERING, February 1985, pp 53-60.

5- Richard Pickvance," A single chip digital signal processor , part 2: Instructions and algorithms, "ELECTRONIC ENGENEERING , March 1985, pp 55-67.

6- Texas instruments, " 16/16 bit : TMS320 family , " ELECTRONIC DESIGN, October 14,1982, pp 150-158.

7- Daniel Essig, Cole Erskine, Edward Caudel and Surendar Magar "A second -generation on Digital Signal Processor, "IEEE JOURNAL OF SOLID- STATE CIRCUITS., Vol-SC-21,No.1,February 1986, pp 86-91.

8- Dave Bursky ," Signal - Processing chips, " ELECTRONIC DESIGN, May 17, 1984, pp 99-109.

9- Texas Instruments, " TMS32020 USER'S GUIDE, Digital Signal Processor products", 1986.

10- Texas Instruments, "Digital Signal Processing Applications with the TMS320 Family", Theory, Algorithms and implementation, 1986.

11- "ANALOG DEVICES" IC Data Book, page 3-237.

12- "Burr-Brown" IC Data Book, Vol.33, page 6.1-53.

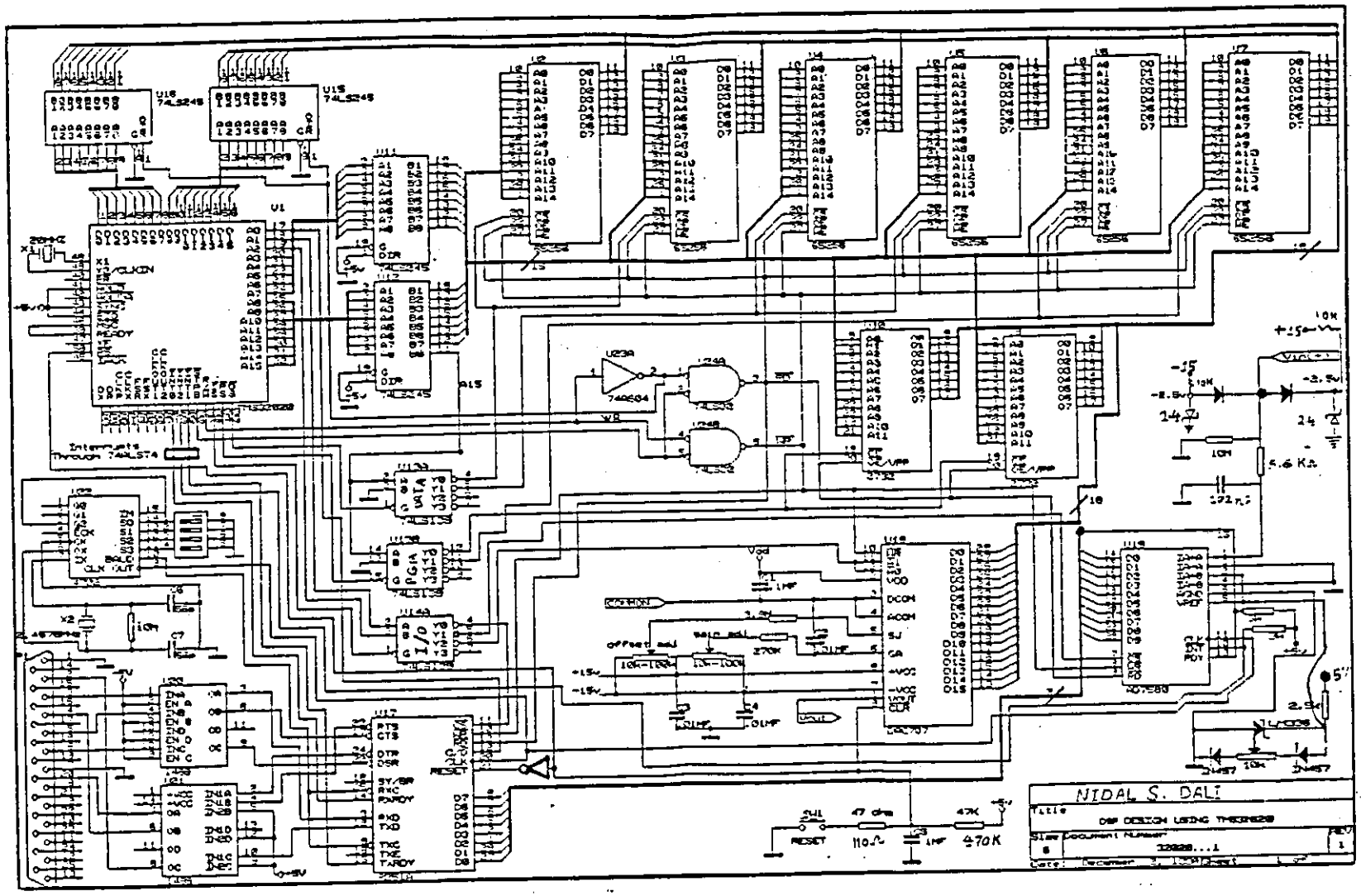
13- Richard Pickvance," A single chip digital signal processor , part 3 : Input / output methods, "ELECTRONIC ENGINEERING, April, 1985, PP 87-95.

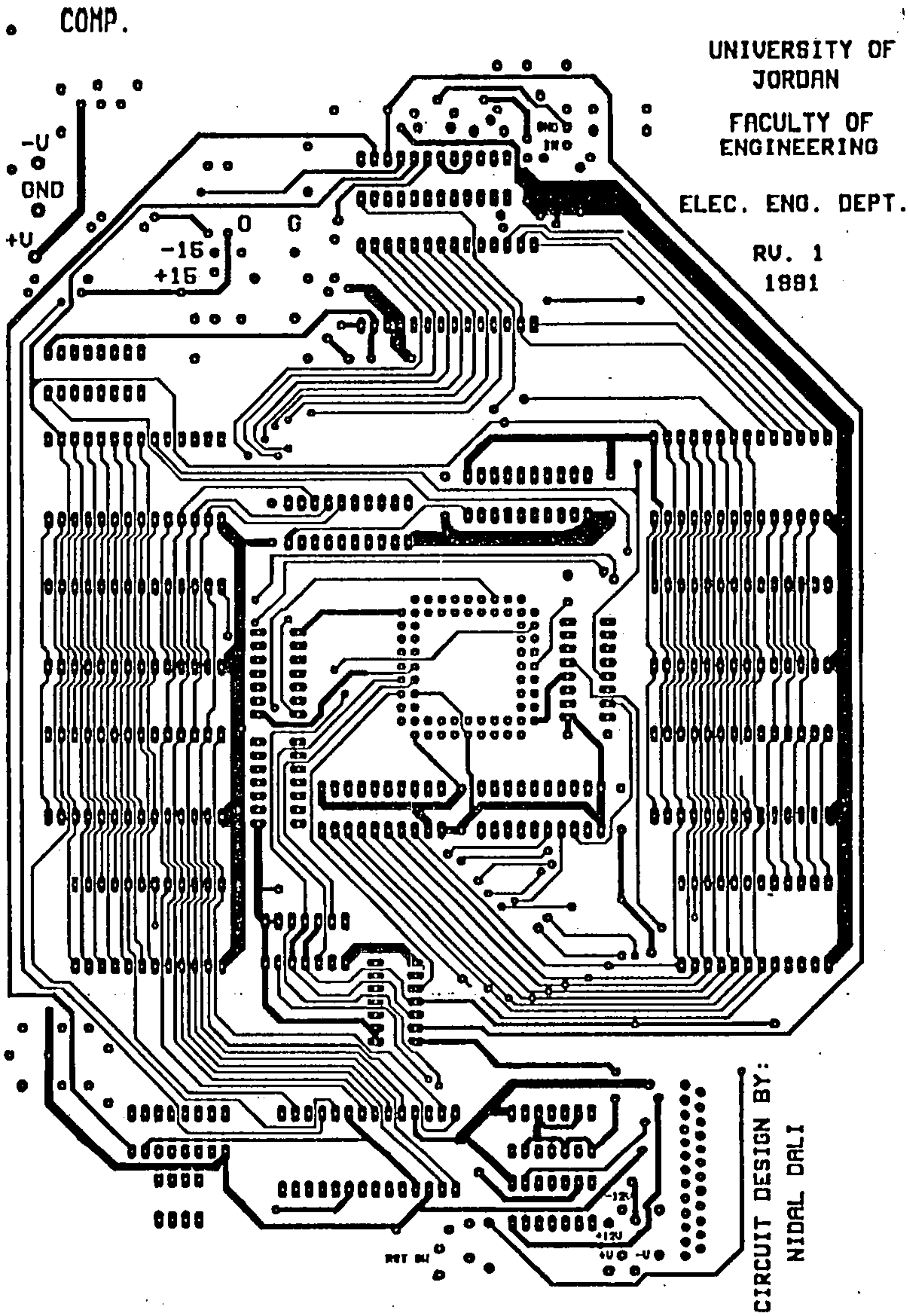
14- "RS data sheets ", July 1984, page 4636 .

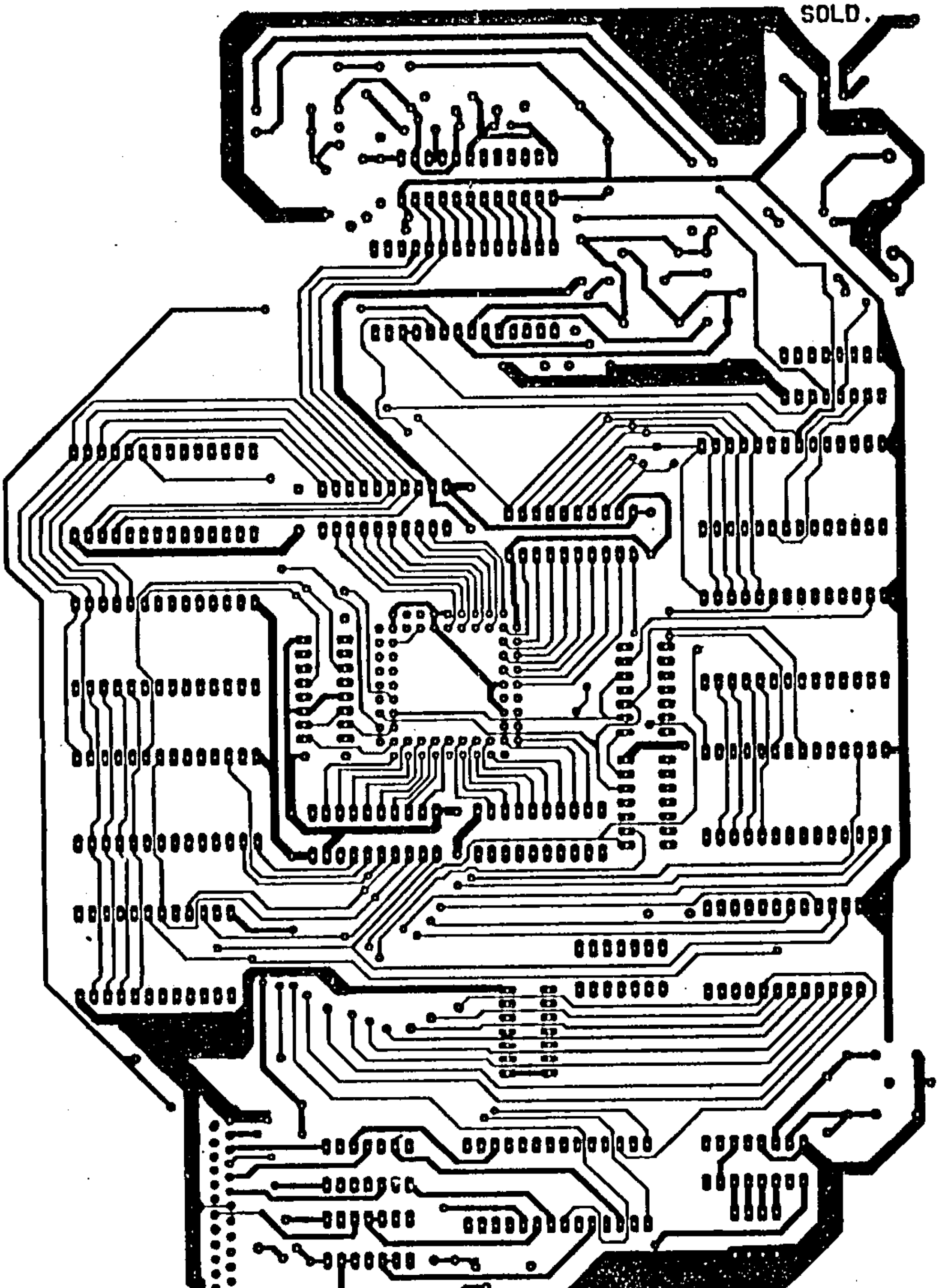
15- J. L. Schmalzel, D.N.Hein, and N.Ahmed. "Some Pedagogical Considarations of Digital Filter Hardware Implementation", IEEE CIRCUTS AND SYSTEMS , Vol.2, No.1, April, 1984, pp 4-9.

16- Winthrop W. and Smith Jr,"Agile development system, running on PC's, builds TMS320- based FIR filter, "ELECTRONIC DESIGN, , June 6,1985, pp 173-181.

17- Ariel Corporation ,"DSP-16 Real- Time Datat Acquisition Processor" for IBM PC/XT/AT, 1986.







```
Ariel TMS32020 Development System  ---------- Mon 4/27/92 0:41am - Pg 1 ----

                          ;                    University of Jordan
                          ;                Faculty of Engineering & Technology
                          ;                -----------------------------------
                          ;
                          ;   This is a sourse file of a program written by N.Dali for
                          ;  Digital Signal Processing board.
                          ;
                          ; Initialization of TMS32020 processor:-
                          ;-------------------------------------------------------------
                                  ORG   32
0020 CE02                 INTMS   ROVM                ;Disable overflow mode.
0021 CE06                         RSXM                ;Suppress sign extension.
0022 C800                         LDPK  0             ;DP Pointer is set to page 0.
0023 CE08                         SPM   0             ;Zero shift product register.
0024 558C                         LARP  AR4
0025 C403                         LARK  AR4,3
0026 D001FFFF                     LALK  65535
0028 60A0                         SACL  *+
0029 60A0                         SACL  *+
002A 6880                         SACH  *
                          ;-------------------------------------------------------------
                          ;Initialization of communication port (UART).
                          ;-------------------------------------------------------------
002B 5580                 INUART  LARP  AR0
002C D0000060                     LRLK  AR0,96
002E CA80                         LACK  128
002F 6080                         SACL  *
0030 E180                         OUT   *,PA1
0031 CA00                         LACK  0
0032 6080                         SACL  *
0033 E180                         OUT   *,PA1
0034 CA00                         LACK  0
0035 6080                         SACL  *
0036 E180                         OUT   *,PA1
0037 CA40                         LACK  64
0038 6080                         SACL  *
0039 E180                         OUT   *,PA1           ;Reset Mode
003A CA4E                         LACK  78
003B 6080                         SACL  *
003C E180                         OUT   *,PA1           ;Mode word
003D CA27                         LACK  39
003E 6080                         SACL  *
003F E180                         OUT   *,PA1           ;Command word
                          ;-------------------------------------------------------------
                          ;TRANSMIT HELP MENU TO PC
                          ;-------------------------------------------------------------
0040 FE8000BB             LOOP1   CALL  CHKRX
0042 8080                         IN    *,0
0043 CA0D                         LACK  13        ;Check if the comming signal is CR
0044 1080                         SUB   *
0045 F5800040                     BNZ   LOOP1
0047 D1000144             TRSHLP  LRLK  AR1,HELP
0049 FE8000C9             LOOP2   CALL  CHKTX
004B 5589                         LARP  AR1
004C 2888                         LAC   *,8,AR0
004D 6880                         SACH  *
004E E080                         OUT   *,0
004F FE8000C9                     CALL  CHKTX
```

```
0051 5589          59          LARP   AR1
0052 E080          60          OUT    *,0
0053 D0012003      61          LALK   8195
0055 10A0          62          SUB    *+
0056 F680005A      63          BZ     CHOISE
0058 FF800049      64          B      LOOP2
                   65 ;Changing the input into upper case
005A FE8000BB      66  CHOISE CALL   CHKRX
005C 8080          67          IN     *,0
005D 2080          68          LAC    *
005E D00400DF      69          ANDK   223
0060 6080          70          SACL   *
                   71 ;Analizing the comming character
0061 CA48          72  COMPAR LACK   72        ;Check for H
0062 1080          73          SUB    *
0063 F6800040      74          BZ     LOOP1
0065 CA4C          75          LACK   76        ;Check for L
0066 1080          76          SUB    *
0067 F680007C      77          BZ     LOAD
0069 CA52          78          LACK   82        ;Check for R
006A 1080          79          SUB    *
006B F680008A      80          BZ     RUN
006D CA44          81          LACK   68        ;Check for D
006E 1080          82          SUB    *
006F F6800098      83          BZ     DSPLAY
0071 CA46          84          LACK   70        ;Check for F
0072 1080          85          SUB    *
0073 F68000B7      86          BZ     FIL
0075 FE8000C9      87  WRONG   CALL   CHKTX    ;Wrong key press
0077 CA07          88          LACK   7         ;Call bell
0078 6080          89          SACL   *
0079 E080          90          OUT    *,0
007A FF800047      91          B      TRSHLP
                   92 ;-------------------------------------------------------------
007C FE8000F9      93  LOAD    CALL   FILL1
007E D2000065      94  THPOSE  LRLK   AR2,101
0080 D3000070      95          LRLK   AR3,112
0082 558A          96  LOOP3   LARP   AR2
0083 2080          97          LAC    *
0084 59AB          98          TBLW   *+,AR3
0085 5590          99          MAR    *-
0086 FB900082     100          BANZ   LOOP3
0088 FF800047     101          B      TRSHLP
                  102 ;-------------------------------------------------------------
008A FE8000DC     103  RUN     CALL   PRPARE
008C D0000061     104          LRLK   AR0,97
008E D1000004     105          LRLK   AR1,4
0090 FE800123     106          CALL   ASTHEX
0092 FE8000D2     107          CALL   ORGADD
0094 2080         108          LAC    *
0095 CE24         109          CALA
0096 FF800047     110          B      TRSHLP
                  111 ;-------------------------------------------------------------
0098 FE8000DC     112  DSPLAY CALL   PRPARE
009A D0000061     113          LRLK   AR0,97
009C D1000009     114          LRLK   AR1,9
009E FE800123     115          CALL   ASTHEX
00A0 FE8000D2     116          CALL   ORGADD
```

```
00A2 FE8000EF    117          CALL  ORGLNG
00A4 D2000065    118          LRLK  AR2,101
00A6 D3000070    119          LRLK  AR3,112
00A8 FE8000C9    120  DD1     CALL  CHKTX
00AA 558A        121          LARP  AR2
00AB 2888        122          LAC   *,8,AR0
00AC 6880        123          SACH  *
00AD E080        124          OUT   *,0
00AE FE8000C9    125          CALL  CHKTX
00B0 558A        126          LARP  AR2
00B1 E3A0        127          OUT   *+,AR3
00B2 5590        128          MAR   *-
00B3 FB9000A8    129          BANZ  DD1
00B5 FF800040    130          B     LOOP1
                 131 ;-----------------------------------------------------------------
00B7 FE8000F9    132  FIL     CALL  FILL1
00B9 FF800047    133          B     TRSHLP
                 134 ;-----------------------------------------------------------------
00BB 5588        135  CHKRX   LARP  AR0
00BC D0000060    136          LRLK  AR0,96
00BE CA02        137          LACK  2
00BF 8180        138          IN    *,1
00C0 4E80        139          AND   *
00C1 F68000BB    140          BZ    CHKRX
00C3 CA30        141          LACK  48
00C4 8180        142          IN    *,1
00C5 4E80        143          AND   *
00C6 F5800047    144          BNZ   TRSHLP
00C8 CE26        145          RET
                 146 ;-----------------------------------------------------------------
00C9 5588        147  CHKTX   LARP  AR0
00CA D0000060    148          LRLK  AR0,96
00CC CA81        149          LACK  129
00CD 8180        150          IN    *,1
00CE 4C80        151          XOR   *
00CF F58000C9    152          BNZ   CHKTX
00D1 CE26        153          RET
                 154 ;-----------------------------------------------------------------
00D2 5588        155  ORGADD  LARP  AR0
00D3 D0000060    156          LRLK  AR0,96
00D5 CA00        157          ZAC
00D6 0CA0        158          ADD   *+,12
00D7 08A0        159          ADD   *+,8
00D8 04A0        160          ADD   *+,4
00D9 00A0        161          ADD   *+
00DA 6080        162          SACL  *
00DB CE26        163          RET                ;Address is stored in location >0065
                 164 ;-----------------------------------------------------------------
00DC 5588        165  PRPARE  LARP  AR0
00DD D0000060    166          LRLK  AR0,96
00DF CA02        167  L1      LACK  2
00E0 8180        168          IN    *,1
00E1 4E80        169          AND   *
00E2 F68000DF    170          BZ    L1
00E4 CA30        171          LACK  48
00E5 8180        172          IN    *,1
00E6 4E80        173          AND   *
00E7 F5800047    174          BNZ   TRSHLP
```

```
00E9 CA0D        175          LACK  13
00EA 8080        176          IN    *,0
00EB 1080        177          SUB   *
00EC F5A000DF    178          BNZ   L1,*+
00EE CE26        179          RET
                 180 ;-----------------------------------------------------------------
00EF 5588        181  ORGLNG  LARP  AR0
00F0 D0000066    182          LRLK  AR0,102
00F2 CA00        183          ZAC
00F3 0CA0        184          ADD   *+,12
00F4 08A0        185          ADD   *+,8
00F5 04A0        186          ADD   *+,4
00F6 00A0        187          ADD   *+
00F7 6080        188          SACL  *
00F8 CE26        189          RET                 ;Length is stored in location >0070
                 190 ;-----------------------------------------------------------------
00F9 FE8000DC    191  FILL1   CALL  PRPARE
00FB D0000061    192          LRLK  AR0,97
00FD D1000009    193          LRLK  AR1,9
00FF FE800123    194          CALL  ASTHEX
0101 FE8000D2    195          CALL  ORGADD
0103 FE8000EF    196          CALL  ORGLNG
0105 D100013F    197          LRLK  AR1,Y
0107 D2000065    198          LRLK  AR2,101       ;AR2 refers to address
0109 D3000070    199          LRLK  AR3,112       ;AR3 refers to length
010B FE8000C9    200  F3      CALL  CHKTX
010D CA0D        201          LACK  13
010E 5589        202          LARP  AR1
010F E080        203          OUT   *,0
0110 10A0        204          SUB   *+
0111 F580010B    205          BNZ   F3
0113 FE8000BB    206  F4      CALL  CHKRX
0115 558A        207          LARP  AR2
0116 80A0        208          IN    *+,0
0117 FE8000BB    209          CALL  CHKRX
0119 558A        210          LARP  AR2
011A 8080        211          IN    *,0
011B CA00        212          ZAC
011C 0090        213          ADD   *-
011D 0880        214          ADD   *,8
011E 60AB        215          SACL  *+,0,AR3
011F 5590        216          MAR   *-
0120 FB900113    217          BANZ  F4
0122 CE26        218          RET
                 219 ;-----------------------------------------------------------------
0123 5588        220  ASTHEX  LARP  AR0           ;Changing ASCII chracter into HEX
0124 2080        221          LAC   *
0125 D00400DF    222          ANDK  223           ;Change into upper case
0127 6080        223          SACL  *
0128 D0040030    224          ANDK  48
012A F6800135    225          BZ    ALPHA
012C 2080        226          LAC   *
012D D004000F    227          ANDK  15
012F 61A0        228          SACL  *+,AR1
0130 5590        229          MAR   *-
0131 FB900123    230          BANZ  ASTHEX
0133 FF80013E    231          B     RETT
0135 2080        232  ALPHA   LAC   *
```

```
0136 D004000F    233          ANDK  15
0138 D0020009    234          ADLK  9
013A 61A0        235          SACL  *+,AR1
013B 5590        236          MAR   *-
013C FB900123    237          BANZ  ASTHEX
013E CE26        238  RETT    RET
                 239 ;-----------------------------------------------------------
013F 4C4F4144    240  Y       DB    'LOAD'
0141 44415441    241          DB    'DATA'
0143 0D20        242          DB     13
0144 0A0A        243  HELP    DB     10,10
0145 20202020    244          DB    '    '
0147 20202020    245          DB    '    '
0149 20202020    246          DB    '    '
014B 20205765    247          DB    '  We'
014D 6C636F6D    248          DB    'lcom'
014F 20746F20    249          DB    ' to '
0151 44696769    250          DB    'Digi'
0153 74616C20    251          DB    'tal '
0155 5369676E    252          DB    'Sign'
0157 616C2050    253          DB    'al P'
0159 726F6365    254          DB    'roce'
015B 7373696E    255          DB    'ssin'
015D 67205379    256          DB    'g Sy'
015F 7374656D    257          DB    'stem'
0161 20445350    258          DB    ' DSP'
0163 36342020    259          DB    '64  '
0165 0D0A0A20    260          DB     13,10,10
0167 20202020    261          DB    '    '
0169 20202020    262          DB    '    '
016B 20202020    263          DB    '    '
016D 20202020    264          DB    '    '
016F 20202020    265          DB    '    '
0171 20202020    266          DB    '    '
0173 20205573    267          DB    '  Us'
0175 696E6720    268          DB    'ing '
0177 544D5333    269          DB    'TMS3'
0179 32303230    270          DB    '2020'
017B 2050726F    271          DB    ' Pro'
017D 63657373    272          DB    'cess'
017F 6F722020    273          DB    'or  '
0181 0D0A0A20    274          DB    13,10,10
0183 20202020    275          DB    '    '
0185 20202020    276          DB    '    '
0187 20202020    277          DB    '    '
0189 20202020    278          DB    '    '
018B 20202020    279          DB    '    '
018D 20202020    280          DB    '    '
018F 20202020    281          DB    '    '
0191 446F6E65    282          DB    'Done'
0193 20427920    283          DB    ' By '
0195 4E696461    284          DB    'Nida'
0197 6C204461    285          DB    'l Da'
0199 6920        286          DB    'i '
019A 0D0A0A0A    287          DB     13,10,10,10
019C 0A20        288          DB     10
019D 20202020    289          DB    '    '
019F 50524553    290          DB    'PRES'
```

```
01A1 533A2D20    291        DB    'S:- '
01A3 0D0A0A20    292        DB     13,10,10
01A5 20312D48    293        DB    ' 1-H'
01A7 20202020    294        DB    '    '
01A9 20202020    295        DB    '    '
01AB 20202020    296        DB    '    '
01AD 2046GF72    297        DB    ' For'
01AF 2048656C    298        DB    ' Hel'
01B1 20702E20    299        DB    ' p. '
01B3 0D0A0A20    300        DB    13,10,10
01B5 20322D4C    301        DB    ' 2-L'
01B7 28616161    302        DB    '(naa'
01B9 612C6C6C    303        DB    'a,ll'
01BB 6C6C2920    304        DB    'll) '
01BD 20466F72    305        DB    ' For'
01BF 204C6F61    306        DB    ' Loa'
01C1 64696E67    307        DB    'ding'
01C3 20612050    308        DB    ' a P'
01C5 6F677261    309        DB    'ogra'
01C7 6D20696E    310        DB    'm in'
01C9 746F2074    311        DB    'to t'
01CB 68652044    312        DB    'he D'
01CD 53502053    313        DB    'SP S'
01CF 79737465    314        DB    'yste'
01D1 6D2E        315        DB    'm.'
01D2 0D0A0A20    316        DB    13,10,10
01D4 20332D52    317        DB    ' 3-R'
01D6 28616161    318        DB    '(naa'
01D8 61292020    319        DB    'a)  '
01DA 20202020    320        DB    '    '
01DC 2046GF72    321        DB    ' For'
01DE 2052756E    322        DB    ' Run'
01E0 6E696E67    323        DB    'ning'
01E2 20612050    324        DB    ' a P'
01E4 726F6772    325        DB    'rogr'
01E6 616D2069    326        DB    'am i'
01E8 6E207468    327        DB    'n th'
01EA 65204453    328        DB    'e DS'
01EC 50205379    329        DB    'P Sy'
01EE 7374656D    330        DB    'stem'
01F0 2E0D0A0A    331        DB    '.',13,10,10
01F2 2020342D    332        DB    '  4-'
01F4 44286161    333        DB    'D(aa'
01F6 61612C6C    334        DB    'aa,l'
01F8 6C6C6C29    335        DB    'lll)'
01FA 2020466F    336        DB    '  Fo'
01FC 72204469    337        DB    'r Di'
01FE 73706C61    338        DB    'spla'
0200 696E6720    339        DB    'ing '
0202 636F6E74    340        DB    'cont'
0204 656E7473    341        DB    'ents'
0206 206F6620    342        DB    ' of '
0208 4D656D6F    343        DB    'Memo'
020A 72792E20    344        DB    'ry. '
020C 0D0A0A20    345        DB     13,10,10
020E 20352D46    346        DB    ' 5-F'
0210 28616161    347        DB    '(naa'
0212 612C6C6C    348        DB    'a,ll'
```

```
0214 6C6C2920    349        DB    '11) '
0216 2046GF72    350        DB    ' For'
0218 2046696C    351        DB    ' Fil'
021A 6C696E67    352        DB    'ling'
021C 20446174    353        DB    ' Dat'
021E 6120696E    354        DB    'a in'
0220 746F2074    355        DB    'to t'
0222 6865204D    356        DB    'he M'
0224 656D6F72    357        DB    'emor'
0226 792E        358        DB    'y.'
0227 0D0A0A20    359        DB     13,10,10
0229 20202020    360        DB    '    '
022B 20202020    361        DB    '    '
022D 20202020    362        DB    '    '
022F 20202020    363        DB    '    '
0231 596F7572    364        DB    'Your'
0233 2043686F    365        DB    ' Cho'
0235 6973653A    366        DB    'ise:'
0237 2D202020    367        DB    '-   '
0239 0320        368        DB    3
                 369
                 370
                 371
                 372
                 373
```

No errors found. 538 words assembled from 373 lines.

```
Ariel TMS32020 Development System  ---------------- Sun 1/3/93 12:30am - Pg 1
-------------------------------------------------

                      1  ;This is a test program to communicate the DSP board
                      2  ;with the PC.
0000 FF800020         3            B       32
                      4            ORG     32
0020 5588             5   INUART   LARP    AR0
0021 D0000060         6            LRLK    AR0,96
0023 CA80             7            LACK    128
0024 6080             8            SACL    *
0025 E180             9            OUT     *,PA1
0026 CA00            10            LACK    0
0027 6080            11            SACL    *
0028 E180            12            OUT     *,PA1
0029 CA00            13            LACK    0
002A 6080            14            SACL    *
002B E180            15            OUT     *,PA1
002C CA40            16            LACK    64
002D 6080            17            SACL    *
002E E180            18            OUT     *, PA1      ;RESET MODE
002F CA40            19            LACK    64
0030 6080            20            SACL    *
0031 E180            21            OUT     *,PA1       ;RESET
0032 CA40            22            LACK    64
0033 6080            23            SACL    *
0034 E180            24            OUT     *,PA1       ;RESET
0035 CA40            25            LACK    64
0036 6080            26            SACL    *
0037 E180            27            OUT     *,PA1       ;RESET
0038 CA40            28            LACK    64
0039 6080            29            SACL    *
003A E180            30            OUT     *,PA1       ;RESET
003B CA4E.           31            LACK    78
003C 6080            32            SACL    *
003D E180            33            OUT     *,PA1       ;MODE WORD
003E CA37            34            LACK    55
003F 6080            35            SACL    *
0040 E180            36            OUT     *,PA1       ;COMMAND WORD
0041 CA37            37            LACK    55
0042 6080            38            SACL    *
0043 E180            39            OUT     *,PA1       ;COMMAND WORD
0044 CA81            40 LOOP1      LACK    129
0045 8180            41            IN      *,PA1
0046 4E80            42            AND     *
0047 6080            43            SACL    *
0048 CA81            44            LACK    129
0049 1080            45            SUB     *
004A F5800044        46            BNZ     LOOP1
004C CA44            47            LACK    68
004D 6080            48            SACL    *
004E E080            49            OUT     *,PA0       ;OUT D
004F CBFF            50            RPTK    255
0050 5500            51            NOP
0051 CA81            52  LOOP2     LACK    129
0052 8180            53            IN      *,PA1
0053 4E80            54            AND     *
0054 6080            55            SACL    *
0055 CA81            56            LACK    129
0056 1080            57            SUB     *
0057 F5800051        58            BNZ     LOOP2
```

```
0059 CA41          59          LACK   65
005A 6080          60          SACL   *
005B E080          61          OUT    *,PA0      ;OUT A
005C CBFF          62          RPTK   255
005D 5500          63          NOP
005E CA81          64  LOOP3   LACK   129
005F 8180          65          IN     *,PA1
0060 4E80          66          AND    *
0061 6080          67          SACL   *
0062 CA81          68          LACK   129
0063 1080          69          SUB    *
0064 F580005E      70          BNZ    LOOP3
0066 CA4C          71          LACK   76
0067 6080          72          SACL   *
0068 E080          73          OUT    *,PA0      ;OUT L
0069 CBFF          74          RPTK   255
006A 5500          75          NOP
006B CA81          76  LOOP4   LACK   129
006C 8180          77          IN     *,PA1
006D 4E80          78          AND    *
006E 6080          79          SACL   *
006F CA81          80          LACK   129
0070 1080          81          SUB    *
0071 F580006B      82          BNZ    LOOP4
0073 CA49          83          LACK   73
0074 6080          84          SACL   *
0075 E080          85          OUT    *,PA0      ;OUT I
0076 CBFF          86          RPTK   255
0077 5500          87          NOP
0078 D0000060      88          LRLK   AR0,96
007A CA80          89          LACK   128
007B 6080          90          SACL   *
007C E180          91          OUT    *,PA1
007D CA00          92          LACK   0
007E 6080          93          SACL   *
007F E180          94          OUT    *,PA1
0080 CA00          95          LACK   0
0081 6080          96          SACL   *
0082 E180          97          OUT    *,PA1
0083 CA40          98          LACK   64
0084 6080          99          SACL   *
0085 E180         100          OUT    *, PA1     ;RESET MODE
0086 CA40         101          LACK   64
0087 6080         102          SACL   *
0088 E180         103          OUT    *,PA1      ;RESET
0089 CA40         104          LACK   64
008A 6080         105          SACL   *
008B E180         106          OUT    *,PA1      ;RESET
008C CA40         107          LACK   64
008D 6080         108          SACL   *
008E E180         109          OUT    *,PA1      ;RESET
008F CA40         110          LACK   64
0090 6080         111          SACL   *
0091 E180         112          OUT    *,PA1      ;RESET
0092 CA4E         113          LACK   78
0093 6080         114          SACL   *
0094 E180         115          OUT    *,PA1      ;MODE WORD
0095 CA37         116          LACK   55
```

```
0096 6080          117              SACL   *
0097 E180          118              OUT    *,PA1       ;COMMAND WORD
0098 CA37          119              LACK   55
0099 6080          120              SACL   *
009A E180          121              OUT    *,PA1       ;COMMAND WORD
009B CA02          122   CHKRX      LACK   2
009C 8180          123              IN     *,1
009D 4E80          124              AND    *
009E F680009B      125              BZ     CHKRX       ;CHECK FOR RXRDY
00A0 CA38          126              LACK   56
00A1 8180          127              IN     *,1
00A2 4E80          128              AND    *
00A3 F58000A6      129              BNZ    ERROR
00A5 8080          130              IN     *,0
00A6 FF800020      131   ERROR      B      32

No errors found.  138 words assembled from 131 lines.
```

Ariel TMS32020 Development System ---------- Sun 7/19/92 0:12am - Pg 1

```
                      1  ;This is a test program to communicate the DSP board
                      2  ;with the PC.
0000 FF800020         3            B       32
                      4            ORG     32
0020 5588             5  INUART   LARP    AR0
0021 D0000060         6            LRLK    AR0,96
0023 CA80             7            LACK    128
0024 6080             8            SACL    *
0025 E180             9            OUT     *,PA1
0026 CA00            10            LACK    0
0027 6080            11            SACL    *
0028 E180            12            OUT     *,PA1
0029 CA00            13            LACK    0
002A 6080            14            SACL    *
002B E180            15            OUT     *,PA1
002C CA40            16            LACK    64
002D 6080            17            SACL    *
002E E180            18            OUT     *, PA1      ;RESET MODE
002F CA40            19            LACK    64
0030 6080            20            SACL    *
0031 E180            21            OUT     *,PA1       ;RESET
0032 CA40            22            LACK    64
0033 6080            23            SACL    *
0034 E180            24            OUT     *,PA1       ;RESET
0035 CA40            25            LACK    64
0036 6080            26            SACL    *
0037 E180            27            OUT     *,PA1       ;RESET
0038 CA40            28            LACK    64
0039 6080            29            SACL    *
003A E180            30            OUT     *,PA1       ;RESET
003B CA4E            31            LACK    78
003C 6080            32            SACL    *
003D E180            33            OUT     *,PA1       ;MODE WORD
003E CA27            34            LACK    39
003F 6080            35            SACL    *
0040 E180            36            OUT     *,PA1       ;COMMAND WORD
0041 CA81            37  LOOP1     LACK    129
0042 B180            38            IN      *,PA1
0043 4E80            39            AND     *
0044 6080            40            SACL    *
0045 CA81            41            LACK    129
0046 1080            42            SUB     *
0047 F5800041        43            BNZ     LOOP1
0049 CA44            44            LACK    68
004A 6080            45            SACL    *
004B E080            46            OUT     *,PA0       ;OUT D
004C CBFF            47            RPTK    255
004D 5500            48            NOP
004E CA81            49  LOOP2     LACK    129
004F B180            50            IN      *,PA1
0050 4E80            51            AND     *
0051 6080            52            SACL    *
0052 CA81            53            LACK    129
0053 1080            54            SUB     *
0054 F580004E        55            BNZ     LOOP2
0056 CA41            56            LACK    65
0057 6080            57            SACL    *
0058 E080            58            OUT     *,PA0       ;OUT A
```

```
0059 CBFF          59              RPTK    255
005A 5500          60              NOP
005B CA81          61  LOOP3       LACK    129
005C 8180          62              IN      *,PA1
005D 4E80          63              AND     *
005E 6080          64              SACL    *
005F CA81          65              LACK    129
0060 1080          66              SUB     *
0061 F580005B      67              BNZ     LOOP3
0063 CA4C          68              LACK    76
0064 6080          69              SACL    *
0065 E080          70              OUT     *,PA0       ;OUT L
0066 CBFF          71              RPTK    255
0067 5500          72              NOP
0068 CA81          73  LOOP4       LACK    129
0069 8180          74              IN      *,PA1
006A 4E80          75              AND     *
006B 6080          76              SACL    *
006C CA81          77              LACK    129
006D 1080          78              SUB     *
006E F5800068      79              BNZ     LOOP4
0070 CA49          80              LACK    73
0071 6080          81              SACL    *
0072 E080          82              OUT     *,PA0       ;OUT I
0073 CBFF          83              RPTK    255
0074 5500          84              NOP
0075 FF800020      85              B       32

No errors found.  89 words assembled from 85 lines.
```

```
Ariel TMS32020 Development System  ---------------- Sun 1/3/93 12:24am - Pg 1
-------------------------------------------------

                        1  ;This is a test program to communicate the DSP board
                        2  ;with the PC.
0000 FF800020           3            B       32
                        4            ORG     32
0020 5588               5   INUART   LARP    AR0
0021 D0000060           6            LRLK    AR0,96
0023 CA80               7            LACK    128
0024 6080               8            SACL    *
0025 E180               9            OUT     *,PA1
0026 CA00              10            LACK    0
0027 6080              11            SACL    *
0028 E180              12            OUT     *,PA1
0029 CA00              13            LACK    0
002A 6080              14            SACL    *
002B E180              15            OUT     *,PA1
002C CA40              16            LACK    64
002D 6080              17            SACL    *
002E E180              18            OUT     *, PA1      ;RESET MODE
002F CA4E              19            LACK    78
0030 6080              20            SACL    *
0031 E180              21            OUT     *,PA1       ;MODE WORD
0032 CA27              22            LACK    39
0033 6080              23            SACL    *
0034 E180              24            OUT     *,PA1       ;COMMAND WORD
0035 CA44              25   DALI     LACK    68
0036 6080              26            SACL    *
0037 E080              27            OUT     *,PA0       ;OUT D
0038 CA41              28            LACK    65
0039 6080              29            SACL    *
003A E080              30            OUT     *,PA0       ;OUT A
003B CA4C              31            LACK    76
003C 6080              32            SACL    *
003D E080              33            OUT     *,PA0       ;OUT L
003E CA31              34            LACK    49
003F 6080              35            SACL    *
0040 E080              36            OUT     *,PA0       ;OUT I
0041 CA00              37            LACK    00
0042 6080              38            SACL    *
0043 E080              39            OUT     *,PA0       ;OUT NULL
0044 FF800035          40            B       DALI
                       41

No errors found.  40 words assembled from 41 lines.
```

```
Ariel TMS32020 Development System ---------- Tue 1/1/80 9:17am - Pg 1
-----------------------------------------------

0000 FF800020          1              B       32
                       2              ORG     32
0020 CE02              3    INITMS    ROVM
0021 CE06              4              RSXM
0022 CB00              5              LDPK    0
0023 CE08              6              SPM     0
0024 558C              7              LARP    AR4
0025 C403              8              LARK    AR4,3
0026 D001FFFF          9              LALK    65535
0028 60A0             10              SACL    *+
0029 60A0             11              SACL    *+
002A 6B80             12              SACH    *
002B CE01             13              DINT
002C FE80004A         14              CALL    INUART
002E FE800060         15              CALL    TxRDY
0030 CA44             16              LACK    68
0031 6080             17              SACL    *
0032 E080             18              OUT     *,PA0          ;OUT D
0033 FE80004A         19              CALL    INUART
0035 FE800060         20              CALL    TxRDY
0037 CA41             21              LACK    65
0038 6080             22              SACL    *
0039 E080             23              OUT     *,PA0          ;OUT A
003A FE80004A         24              CALL    INUART
003C FE800060         25              CALL    TxRDY
003E CA4C             26              LACK    76
003F 6080             27              SACL    *
0040 E080             28              OUT     *,PA0          ;OUT L
0041 FE80004A         29              CALL    INUART
0043 FE800060         30              CALL    TxRDY
0045 CA49             31              LACK    73
0046 6080             32              SACL    *
0047 E080             33              OUT     *,PA0          ;OUT I
0048 FF800020         34              B       32
004A 5588             35    INUART    LARP    AR0
004B D0000060         36              LALK    AR0,96
004D CA80             37              LACK    128
004E 6080             38              SACL    *
004F E180             39              OUT     *,PA1
0050 CA00             40              LACK    0
0051 6080             41              SACL    *
0052 E180             42              OUT     *,PA1
0053 CA00             43              LACK    0
0054 6080             44              SACL    *
0055 E180             45              OUT     *,PA1
0056 CA40             46              LACK    64
0057 6080             47              SACL    *
0058 E180             48              OUT     *,PA1
0059 CA4E             49              LACK    78
005A 6080             50              SACL    *
005B E180             51              OUT     *,PA1
005C CA27             52              LACK    39
005D 6080             53              SACL    *
005E E180             54              OUT     *,PA1
005F CE26             55              RET
0060 CA81             56    TxRDY     LACK    129
0061 8180             57              IN      *,PA1
0062 4E80             58              AND     *
```

```
0063 6080        59              SACL    *
0064 CA81        60              LACK    129
0065 1080        61              SUB     *
0066 F5800060    62              BNZ     TxRDY
0068 CE26        63              RET
                 64

No errors found.  75 words assembled from 64 lines.
```

```
Ariel TMS32020 Development System  ---------- Thu 7/30/92 0:35am - Pg 1 ---
-------------------------------------------

                    1  ;THIS IS A TEST PROGRAM TO TEST THE DAC TO GENERATE A SQUA
RE WAVE.
0000 FF800020       2          B       32
                    3          ORG     32
0020 5588           4          LARP    AR0
0021 D0000060       5          LRLK    AR0,96
0023 D0010000       6  START   LALK    0000
0025 6080           7          SACL    *
0026 CB64           8          RPTK    100
0027 E480           9          OUT     *,PA4
0028 D001FFFF      10          LALK    65535
002A 6080          11          SACL    *
002B CB96          12          RPTK    150
002C E580          13          OUT     *,PA5
002D FF800023      14          B       START

No errors found.  17 words assembled from 14 lines.
```

```
Ariel TMS32020 Development System  ---------------- Wed 8/12/92 0:16am - Pg 1 ---
-------------------------------------------------

                    1  ;THIS IS A TEST PROGRAM TO TEST THE DAC TO GENERATE A SQUA
RE WAVE.
0000 FF800020       2          B       32
                    3          ORG     32
0020 5580           4          LARP    AR0
0021 00000060       5          LALK    600,96
0023 00010000       6  START   LALK    0000
0025 6080           7          SACL    *
0026 CB64           8          RPTK    100
0027 FE8000031      9          CALL    DALI
0029 0001FFFF      10          LALK    65535
002B 6080          11          SACL    *
002C CB96          12          RPTK    150
002D FE800030      13          CALL    DIR
002F FF800023      14          B       START
0031 E480          15  DALI    OUT     *,PA4
0032 5500          16          NOP
0033 5500          17          NOP
0034 5500          18          NOP
0035 5500          19          NOP
0036 5500          20          NOP
0037 5500          21          NOP
0038 5500          22          NOP
0039 5500          23          NOP
003A 5500          24          NOP
003B 5500          25          NOP
003C CE26          26          RET
003D E580          27  DIR     OUT     *,PA5
003E 5500          28          NOP
003F 5500          29          NOP
0040 5500          30          NOP
0041 5500          31          NOP
0042 5500          32          NOP
0043 5500          33          NOP
0044 5500          34          NOP
0045 5500          35          NOP
0046 5500          36          NOP
0047 5500          37          NOP
0048 CE26          38          RET

No errors found.  43 words assembled from 38 lines.
```

```
Ariel TMS32020 Development System  --------------- Sat 8/22/92 0:18am - Pg 1
-------------------------------------------------

                        1  ;THIS IS A TEST PROGRAM TO TEST THE DAC AND THE A/D .
0000 FF800020           2          B       32
                        3          ORG     32
0020 5588               4          LARP    AR0
0021 B0000060           5          LRLK    AR0,96
0023 B0010000           6  START   LALK    0000
0025 6080               7          SACL    *
0026 E280               8          OUT     *,PA2
0027 CB32               9          RPTK    50
0028 5500              10          NOP
0029 8280              11          IN      *,PA2
002A 5500              12          NOP
002B 5500              13          NOP
002C E480              14          OUT     *,PA4
002D CB32              15          RPTK    50
002E 5500              16          NOP
002F FF800023          17          B       START

No errors found.  19 words assembled from 17 lines.
```