The Itanium - 1986 8 Bit Microprocessor Report

By

PRIYANK JAIN
(02010123)

Group # 11

Under guidance of

Dr. J. K. Deka & Dr. S. B. Nair

Department of Computer Science & Engineering
Indian Institute of Technology, Guwahati.
Date : April 21, 2004

TABLE OF CONTENTS 2

Design Requirements :

Characteristics

- 8 bit Data bus.
- 8 bit Address bus.
- 3 general purpose register

Functions Implemented

Memory Access instructions:

1. Load
2. Store

Arithmetic Operations:

3. Add (data)
4. Add (registers)
5. Multiply
6. Subtract (data)
7. Subtract (registers)

Logical Operations:

8. NOT
9. OR
10. AND

Internal Register Access :

11. Move (A->B)
12. Move (B->A)

Control Transfer Instructions:

13. Jump (unconditional)
14. Jump (conditional)
15. Call
16. Return
17. Halt

Design Characteristics :

- Design on a **CISC based architecture**
- Microprogramming logic used.

Control Unit Design

The CPU has a MICRO PROGRAMMED CONTROL UNIT. The design features characteristics state of CISC Architecture. 4 EPROM's(Erasable Programmable Read only memory Chips) were used for this purpose. A total of 32 control signals were used.

- EPROM 1
  - Add7 — ADDRESS LINES TO JUMP TO
  - Add6
  - Add5
  - Add4
  - Add3
  - Add2
  - Add1
  - Add0

- EPROM 2
  - A3 — ADDRESS BITS FOR A DECODER
  - A2
  - A1
  - B3 — ADDRESS BITS FOR B DECODER
  - B2
  - B1
  - MDR (READ)
  - MDR (WRITE)

- EPROM 3
  - C3 — ADDRESS BITS FOR C DECODER
  - C2
  - C1
  - PC++ — SIGNAL TO INCREMENT PC
  - TOS — UP/DOWN SIGNAL FOR SP
  - LE — LATCH ENABLE FOR FLAG LATCH
  - CS — CHIP SELECT(FOR MAIN MEMORY)
  - R/W — READ/WRITE (FOR MAIN MEMORY)

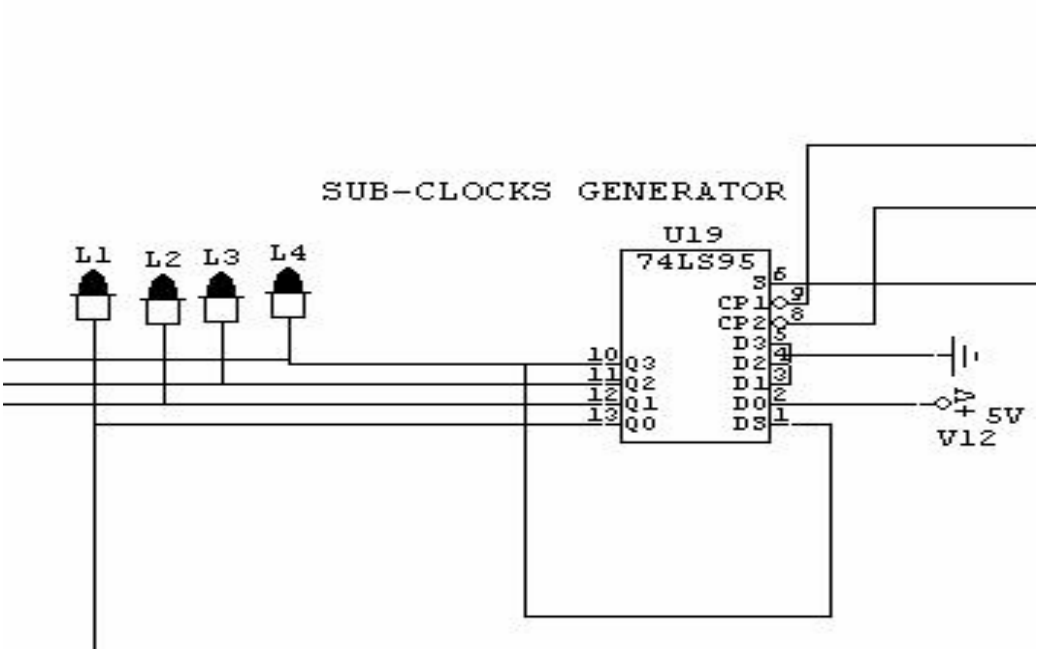
- EPROM 4
  - S3 — 4 SELECT LINES FOR ALU
  - S2
  - S1
  - S0
  - M — 2 ARITHMETIC/LOGIC LINES FOR ALU
  - C
  - P1 — 2 CONDITION BITS
  - P2

Basic Architecture

The basic architecture of the CPU consists of a set of registers (special purpose and general purpose) ,some of which are connected to A and B buses which is in turn connected to the ALU. The arithmetic and logic operations are performed by the ALU. Connected to these buses are the MAR(memory address register) and MDR(memory data register) which are directly connected to the RAM chip 6116(having 8 data lines and 8 address lines).The clock subcycle generator (JHONSON counter) sends clock pulses to all the above entities. The control signals and the select lines are provided by the MICRO INSTRUCTION register (which is connected to the MICRO CONTROL STORE).

Clocking Mechanism

A JHONSON counter has been used to generate 4 subcycles which supply the clock to the entire CPU. The principle of JHONSON counter is that just 1 bit of the 4 bit shift register 7495 is kept high. The highest bit is again fed into the serial input to generate 4 clock pulses with a phase shift.



As can be seen Q3 is fed into Ds. Shift clock is the main clock of the CPU. Discreet waveforms having a phase shift can thus be obtained.
The following actions are performed at each clock sub-cycle

CLOCK SUBCYCLE 1: The latch of the MICRO INSTRUCTION REGISTER is enabled during this clock cycle.

CLOCK SUBCYCLE 2: A & B latches latch on to the values of the A and B bus respectively.

CLOCK SUBCYCLE 3: MEMORY ADDRESS REGISTER takes in the address from the A and B buses.

CLOCK SUBCYCLE 4: All the special purpose and general purpose registers get the 4'th clock sub-cycle.

Memory Interfacing

The Main Memory consists of the RAM chip 6116. The actual size of the memory is 2 X1024 X 8 bits. But in this architecture only 8 address lines are used *ie*. 256 memory locations are accessed. The data bus is used for both load as well as store operations. Tri-state buffers are used so that the same bus can be used for both read and write operations.

The two very important registers used to interface the internal logistics of the CPU with the main memory are -

1) MEMORY ADDRESS REGISTER
2) MEMORY DATA REGISTER

Hardware Requirements

- Primitive IC s (74XXX , RAM, ROM)
- Power Supply
- TTL Logic Clock Input

IC ‘s Used –

1.	SRAM – 6116 (8 bit data bus & 11 bit Address Bus)		X	1
2.	EPROM – INTEL 2732		X	4
3.	ALU – 74181	(4 bit ,16 function)	X	1
4.	74154	(4X16 Decoder)	X	3
5.	7495	(4 bit Shift Register With PL)	X	4
6.	74194	(4 bit Universal Register)	X	4
7.	74191	(4 bit Up / Dn. Counter)	X	1
8.	74161	(4 bit up counter)	X	4
9.	74373	(8 bit latch)	X	11
10.	7475	(4 bit Latch)	X	3
11.	74367	(Hex Tri-state Buffer)	X	4
12.	7404	(Hex Inverter)	X	2
13.	7408	(Quad AND Gate)	X	2
14.	7432	(Quad OR Gate)	X	2
15.	7447	(BCD to 7 Segment Decoder)	X	3
16.	2005	(7 Segment Display)	X	3

Power Supply – +5 V and 0 V D.C Supply for all ICs .

Clock Input – A Frequency generator was used to provide a TTL logic clock. Circuit was tested to be functional for KHz frequency Range.

Instruction Set

The CPU is designed to execute a set of 17 instructions an 8-bit opcode is required. This is the op-code assignment for each instruction.

LOAD	00000010
STORE	00000110
JUMP	00001010
ADD(DATA)	00001110
AND	00010001
CALL	00010111
MOVE(A-B)	00100011
NOT	00100101
SUB(REG)	00100111
JUMP(A=B)	00101010
RETURN	00101101
MULTIPLY	00111011
SUB(DATA)	01000111
ADD(REG)	01001010
MOVE(B->A)	01001100
HALT	01001110
OR	00010100

Table 3.1 Instruction Set

To execute an instruction one needs to give an 8 - bit opcode followed by 4-bit operand.

The CPU has four general purpose registers, thus 2 bits are required to reference one register. Thus using four bits 2 registers can be referenced. The context in which the register is referenced is instruction specific.

LOAD

This is one of the two memory access instructions. This is used to bring the contents of a memory location into the registers of the CPU. Its opcode is 00000010 and the operand is used to specify the address which should be accessed. The content of the memory location is stored into the accumulator

STORE

This is the other instruction that accesses the memory. This is used to store the content of registers into the main memory. The construct is similar to that of LOAD. Its opcode 00000110 and the operand is used to specify the address where the where the data has to be stored. When this instruction is executed the contents of the accumulator is stored into the memory.

JUMP

This is the unconditional jump statement. It is used to set the program counter to a value of the succeeding location. It opcode is 00001010. The operand is used to specify the 8 bit jump address.

ADD (Data)

This is an arithmetic instruction which uses the ALU. Its opcode is 00001110. The operand is used to specify the value to be added to the accumulator. The result of the operation is stored in the accumulator. The carry bit is not handled in this architecture.

AND

This is a logical operation and uses the ALU. Its opcode is 00010001. This performs bitwise AND operation on the content of the accumulator with the operand. The result of the operation is stored in the accumulator

CALL

This is the instruction used to make a subroutine call. When this is done the current value of program counter (PC) is stored in the stack register and then the program counter is set to the value depending on the location to which the call takes place. Its opcode is 00010111.

MOVE (A->B)

This is used to copy the content of register A into B. Its opcode is 00100011. This instruction is very significant as the results of all alu and memory operations are stored in the accumulator.

NOT

This is a logical instruction in which bit wise binary inversion is done. Its opcode is 00100101 . The result of this operation is stored in the accumulator (register A)

SUB (Reg)

This is an arithmetic instruction. Its opcode is 00100111 . The difference is stored in the accumulator. The architecture supports negative numbers.

JUMP (A=B)

This is a the conditional counterpart to the jump statement. In this statement the jump occurs only if the contents of the registers A & B are same. Its opcode is 00101010. The operand is used to specify the address locations to jump to .

RETURN

This is used to return from a subroutine call. When this instruction is executed the contents of the stack register is set into the program counter. Thus the program continues execution from the location following the one from which call took place. Its opcode is 00101101.

MULTIPLY

This is an arithmetic instruction. Its opcode is 00111011. The result is stored in the accumulator. The architecture does not support negative numbers.

SUB (Data)

This is an arithmetic instruction. Its opcode is 0110. The operand is used to specify the value which is to be subtracted from the accumulator. The difference is stored in the accumulator. The architecture does support negative numbers.

ADD (Reg)

This is an arithmetic instruction which uses the ALU. Its opcode is 00001110. The result of the operation is stored in the accumulator. The carry bit is not handled in this architecture.

MOVE (B->A)

This is used to copy the content of one register into another. Its opcode is 01001100. This instruction is very significant as the results of all alu and memory operations are stored in the accumulator.

OR

This is another logical operation. Its opcode is 00010100. This performs bitwise OR operation on the contents of the registers specified by the operand and stores the result of the operation in the accumulator.

HALT

This is the instruction used to signify the end of the program. It opcode is 01001110. After this instruction the clock to the entire CPU is disabled. The state of the registers are frozen and they continue to store the value last held by them. This instruction is irreversible and execution cannot be continued without putting the power back on again.

Design and Implementation

Register Organization

The registers in the architecture can be broadly divided into two categories, special registers and general purpose registers. This classification is by no means mutually exclusive and some registers seem to have characteristics of both especially the accumulator.

Special Registers

Memory Data Register

This is the register which is connected to the data lines of the memory. Any value which is to be written onto the memory first gets stored in the memory data register. Also any data or address to be read from the main memory gets stored in the M.D.R

Memory Address Register

This is the register which is connected to address lines of the main memory. The address location to be referenced is transferred from the bus onto the M.A.R

Program Counter

This is the register which is used to store the address location of the next instruction. Its value is to changed during the JUMP instruction. Also during CALL and RETURN program counter's value is to be stored and retrieved from the stack. It actually consists of two 4 bit registers.

Stack Register

This is the register where the program counter is stored when a CALL instruction takes place. The stack register actually consists of two 4 bit registers. This register stores the program counter. The value stored in this register is used to reset the program counter in the case of RETURN.

Micro Program Counter

This is the register which is used to reference the Micro Control Store(EPROM's).The Instruction from the main memory is directly translated into the address location of the Micro Control Store, where the Micro Program has already been stored.

Micro Instruction Register

All the 32 control signals get latched onto the Micro instruction Register at the 1'st clock pulse. Then the micro instruction register sends all the control signals to the respective decoders and select lines.

Accumulator

This can be considered to be a special general purpose register. This is a register that is available to the programmer space. This is the register where all the results of all ALU operations are stored. This is also the only register which can interact with the RAM memory by means of LOAD and STORE. Assignment operation can also take place only into this register. This register is referred to by A. Accumulator can perform all the functions that can be performed by all other registers and can be used for all manipulations by the programmer.

General Purpose Registers

They are three in number including the accumulator. The programmer can use them for both data storage and manipulation. When the CPU starts these register are set to an arbitary value . They can be used as input to the ALU; either as first operand or second, but the result of the ALU operation comes into the accumulator and has to be shifted to other register by means of MOVE instruction.

Subroutine Support

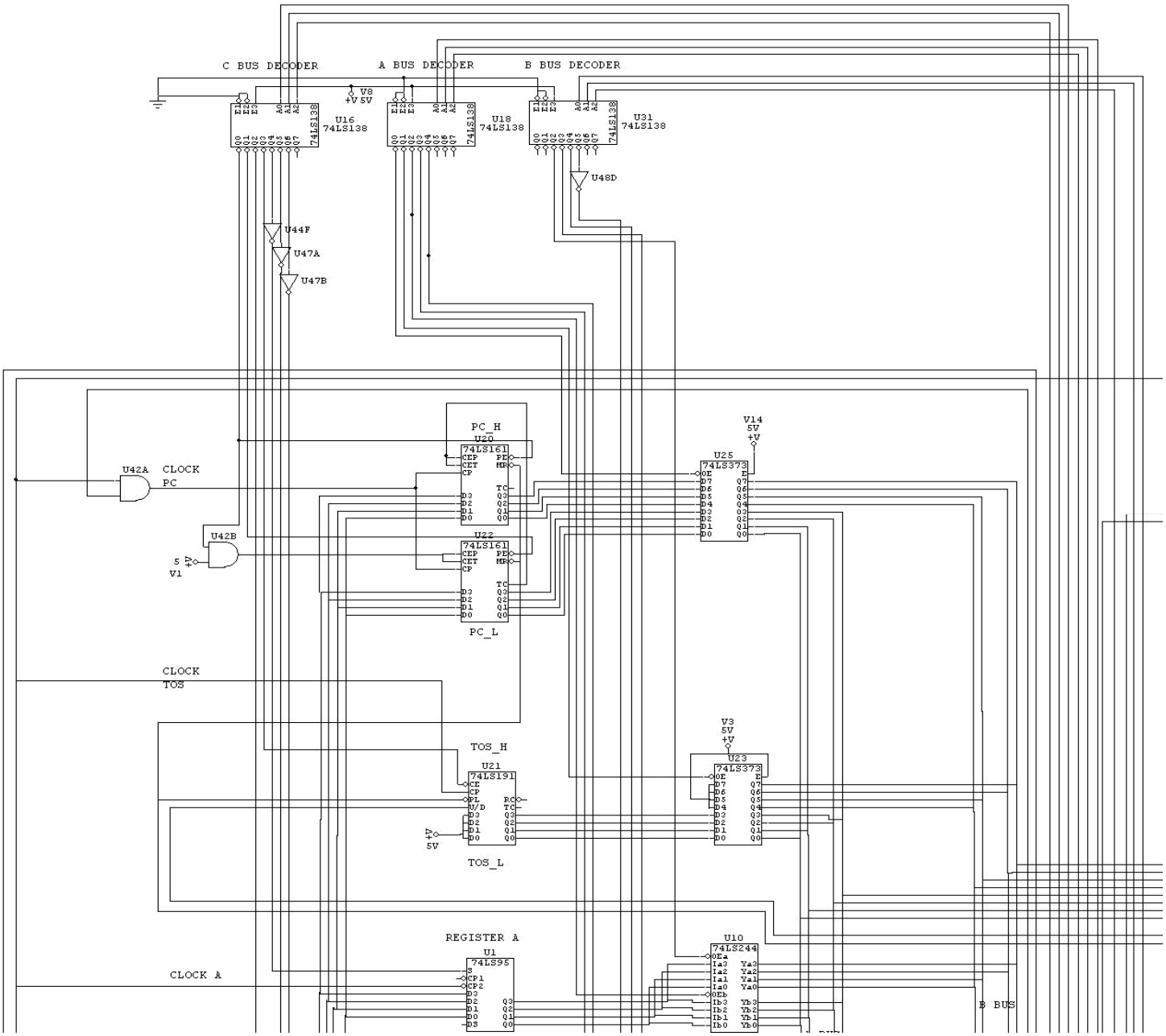
The architecture has built in support for function call. A programmer may make a subroutine call to any location in the program store, execute instructions there and then jump back to the initial location. He is not required to handle to which location he has to jump back to. This is handled by the architecture. More than one levels of subroutine calls is supported. Thus recursion is automatically supported.

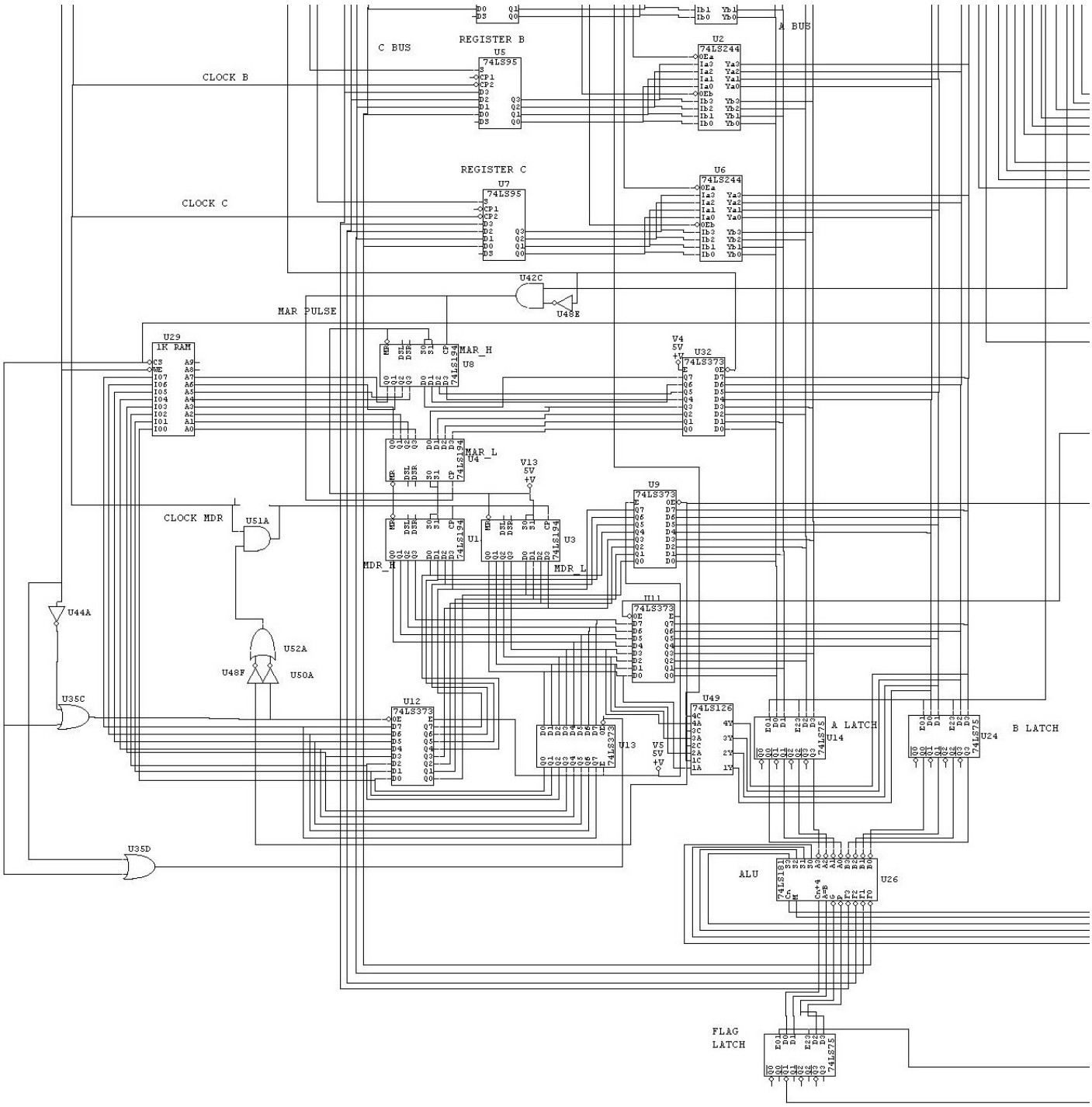
At the heart of the subroutine call support is the two stack registers. These register stores the value of the program counter whenever a function call takes place. This value is set back into the program counter whenever a return statement is encountered.

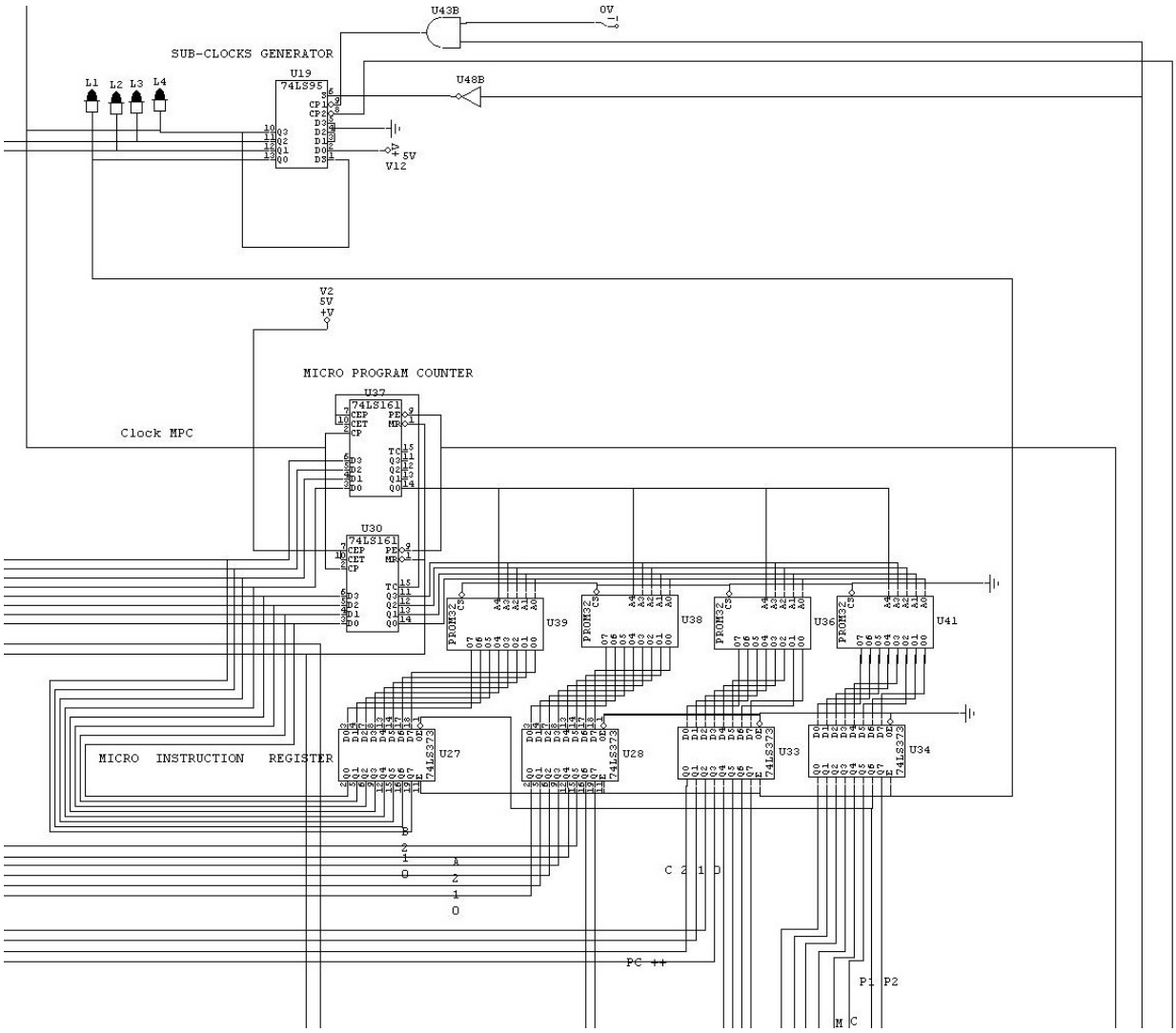
Jump Support

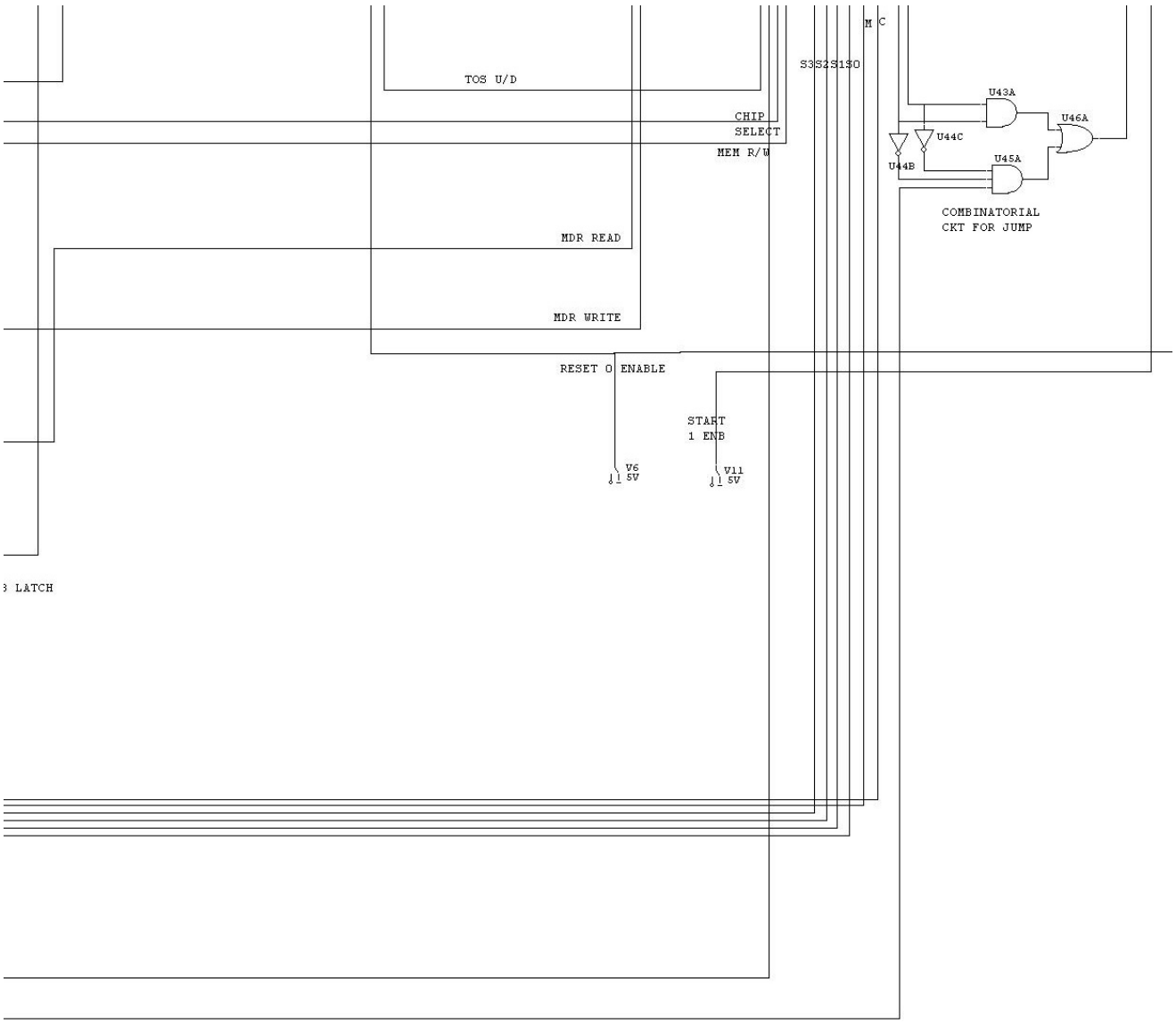
The architecture supports two kinds of jump statements. In both cases the appropriate value as specified in the operands is set in the program counter. There are two types of jumps. An unconditional jump where the jump always takes place and a conditional jump in which case the jump takes place only if the contents of the register A and B are same. For conditional jump a comparator is used. A flag register was maintained and thus ALU output could be used for a jump.

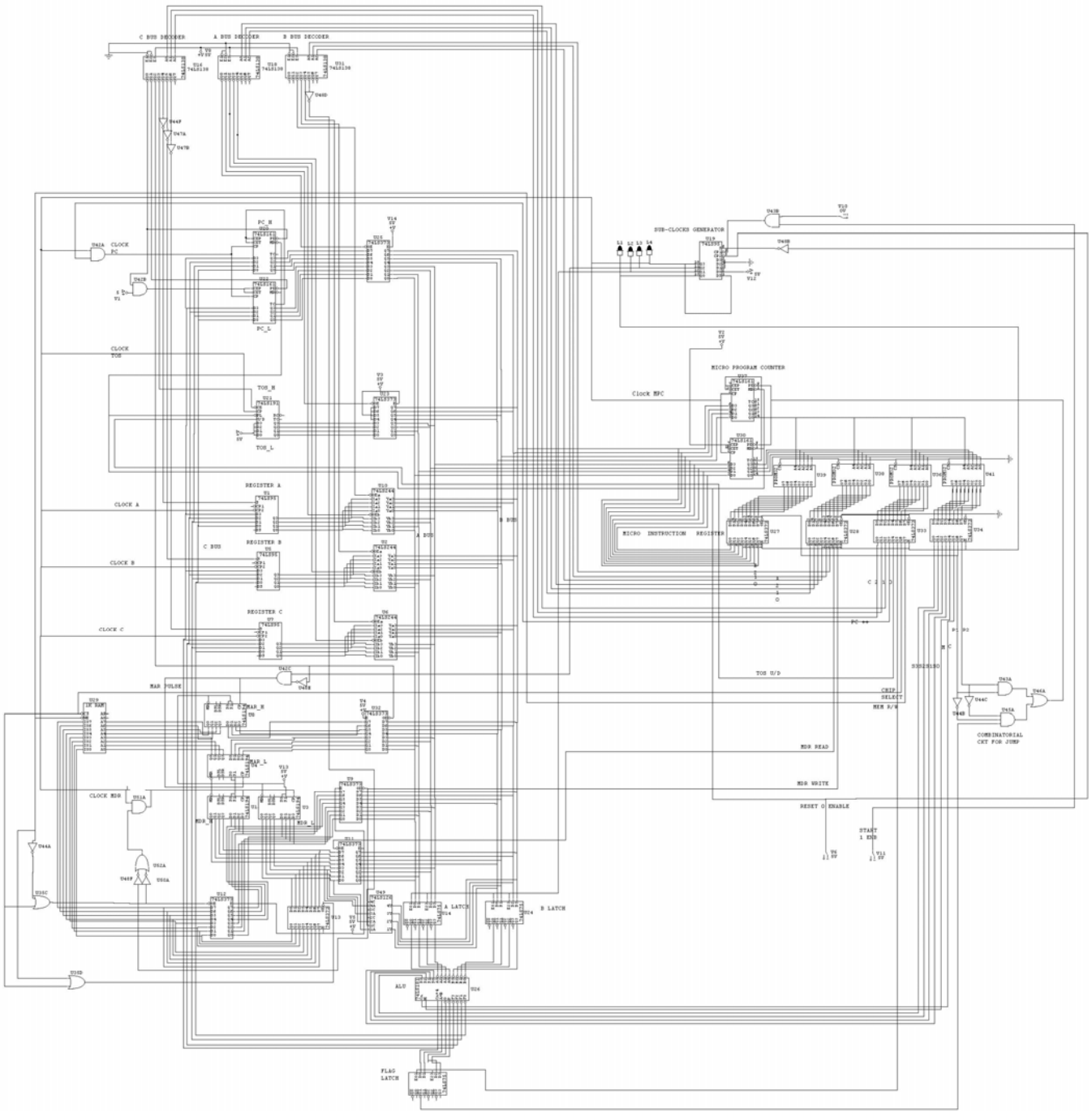
DETAILED CIRCUIT DIAGRAM




C BUS
A BUS
REGISTER B
REGISTER C
CLOCK B
CLOCK C
U5
74LS95
U7
74LS95
U2
74LS244
U6
74LS244
U42C
U48E
MAR PULSE
U29
1K RAM
MAR_H
U8
MAR_L
U4
V4
5V
+V
U32
74LS373
V13
5V
+V
U9
74LS373
CLOCK MDR
U51A
U1
U3
MDR_H
MDR_L
U44A
U52A
U48F
U50A
U35C
U11
74LS373
U12
74LS373
U13
74LS373
V5
5V
+V
U49
74LS126
A LATCH
U14
B LATCH
U24
U35D
ALU
U26
FLAG
LATCH


SUB-CLOCKS GENERATOR
U43B
0V
U48B
L1 L2 L3 L4
U19
74LS395
+5V
V12
V2
5V
MICRO PROGRAM COUNTER
U37
74LS161
U30
74LS161
Clock MPC
PROM32
U39
U38
U36
U41
MICRO INSTRUCTION REGISTER
U27
74LS373
U28
74LS373
U33
74LS373
U34
74LS373
C 2 1 0
PC ++
P1 P2
M C


M C
S3S2S1S0
TOS U/D
CHIP
SELECT
MEM R/W
U43A
U46A
U44C
U44B
U45A
COMBINATORIAL
CKT FOR JUMP
MDR READ
MDR WRITE
RESET O ENABLE
START
1 ENB
V6
5V
V11
5V
3 LATCH


C BUS DECODER
A BUS DECODER
B BUS DECODER
PC_H
PC_L
CLOCK PC
CLOCK TOS
TOS_H
TOS_L
REGISTER A
REGISTER B
REGISTER C
CLOCK A
CLOCK B
CLOCK C
A BUS
B BUS
C BUS
MAR PULSE
MAR_H
MAR_L
CLOCK MDR
MDR_H
MDR_L
A LATCH
B LATCH
ALU
FLAG LATCH
SUB-CLOCKS GENERATOR
MICRO PROGRAM COUNTER
Clock MPC
MICRO INSTRUCTION REGISTER
PC ++
TOS U/D
CHIP SELECT
MEM R/W
COMBINATORIAL CKT FOR JUMP
MDR READ
MDR WRITE
RESET O ENABLE
START 1 ENB