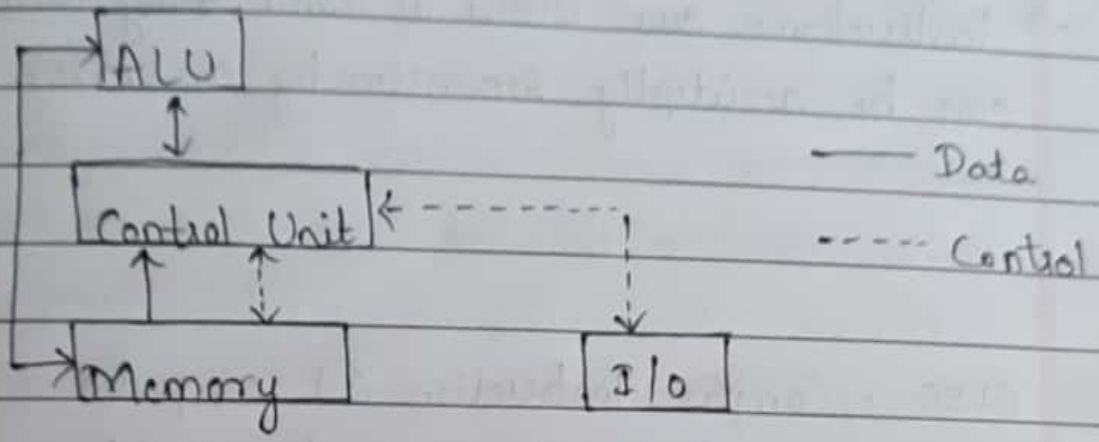
1. Von Neuman Architecture



> All parts are controlled by control unit CPU and connected togather by BUS
> Data can pass through bus in half duplex mode to and from CPU
> Memory holds program and data known as stored program concept.
> Memory is split to small cells with the same size Their original numbers are called address numbers.

Advantages.

Control unit gets data & instruction in the same way from one memory. It simplifies design and development of the control unit.

Disadvantages.

=> Serial Instruction processing doesn't allow parallel execution parallel execution are simulated later by OS

→ one bus is bottle neck
→ Instructions are stored in same ~~way~~ memory as data can be accidently rewritten by an error in a program

2. CISC - Complex Instruction Set computer

CISC was developed to make compiler development easier and simple. CISC is complex Instruction set computer.

RISC

RISC is designed to perform a smaller number of and types of computer instructions. Its a microprocessor that is designed to perform smaller number of computer instruction. RISC is Reduced Instruction Set Computer

CISC	RISC
> More set of Instruction	> Lesser set of Instruction
> More addressing mode	> Lesser addressing mode
> Minimum amount of RAM	> More amount of RAM
> Focus is on hardware to optimise	> Focus is on software to optimize
> High power consumption	> Low power consumption
> Not highly pipelined	> Highly pipelined
> Large code size	> Small code size

RELIABLE
Page No.:
Date : / /

3. There are four main types of numbering system.
> Binary number system (Base-2)
> octal number system (Base-8)
> Decimal number system (Base 10)
> Hexadecimal number system (Base 16)

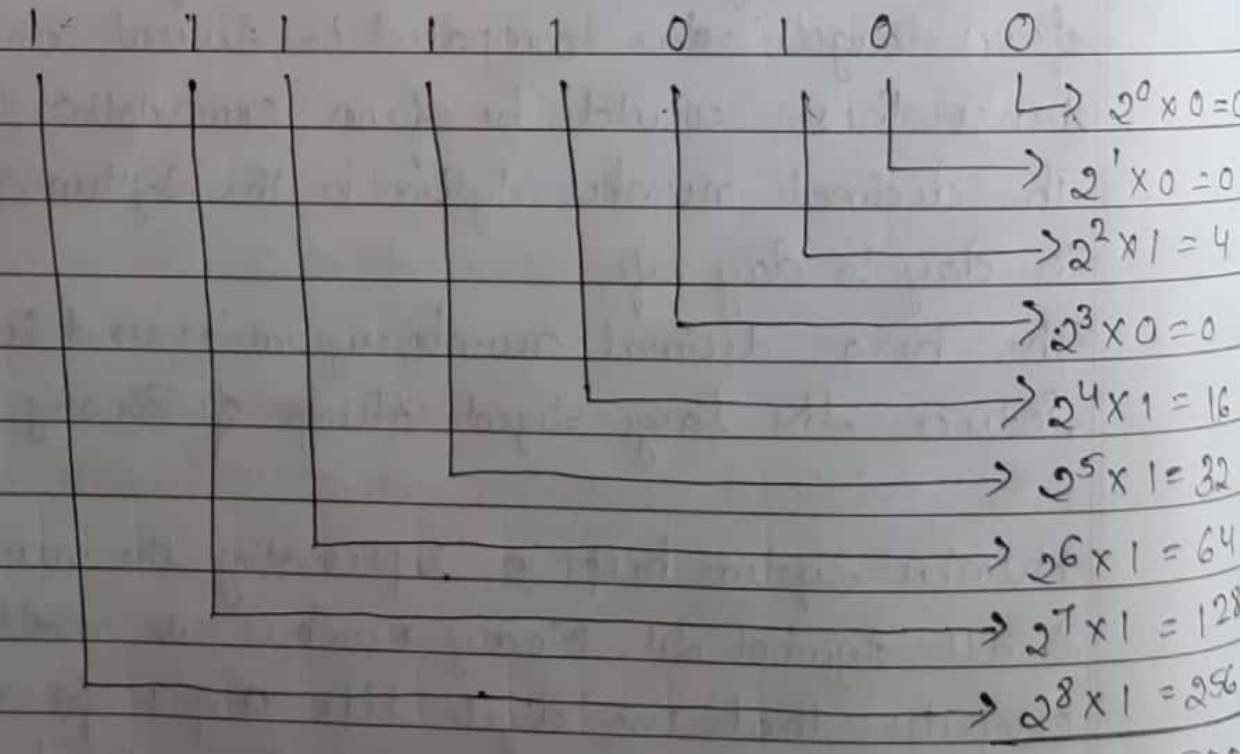
Binary number system is used in operation of computer and to store data in computer.
The advantage of octal number system is that it has fewer digits when compared to several other systems. hence there could be fewer computation error.
The decimal number system is the system that we use in day to day life
The hexa decimal numbersystem is used in computer to reduce the large sized strings of Binary system.

Number system helps in representing the numbers in a small symbol id. Binary numbers are mostly used in computer that use digits like 0 & 1 for calculating simple problems. The number system also help in conversion of one number system to another.

Conversion from one number system to another

To convert a number from one of the binary/octal/hexadecimal systems to one of other systems, we first convert it into decimal and then we convert it to required system by using below process.

Ex: 11111100.



$11111100_{(2)} = 500_{(10)}$ //

RELIABLE
Page No.:
Date : / /

$500_{(10)} = 764_{(8)}$

8	500
8	62 – 4 ↑
	7 – 6
	→

$764_{(8)}$ //

$500_{(10)} = 1F4_{(16)}$

16	500
16	31 – 4 ↑
	1 – 15
	→

15 in hexadiimal is F

∴ $1F4_{(16)}$ //

2	500
2	250 – 0
2	125 – 0
2	62 – 1
2	31 – 0
2	15 – 1
2	7 – 1
2	3 – 1
	1 – 1

$111110100_{(2)}$ //

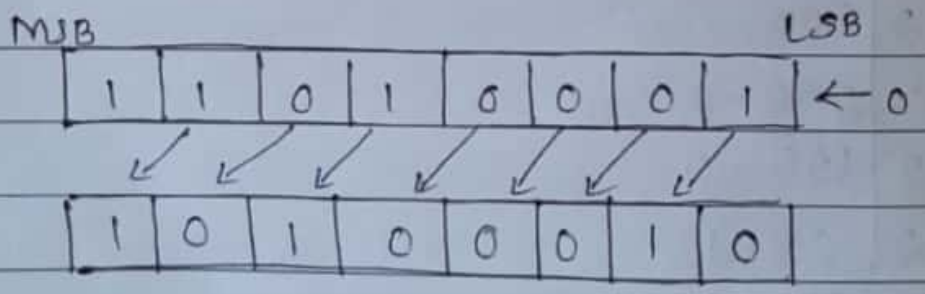
4 Logical shift and Arithmatic shift are bit manipulation operation.

Logical shift.

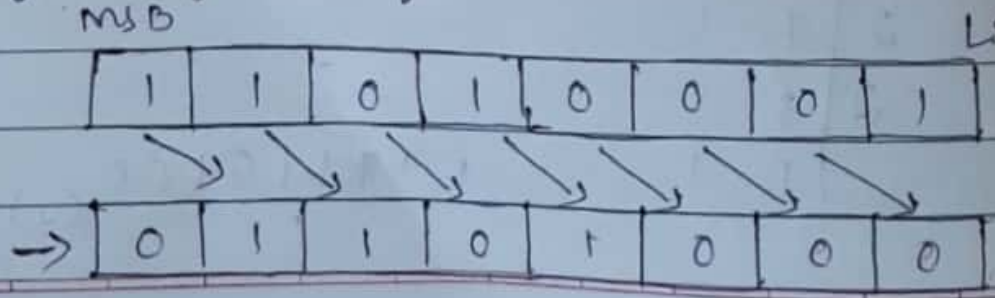
> A left logical shift of one position moves each bit to the left by one. The vacant least significant bit (LSB) bit is filled with zero and the most significant bit MSB is discarded.

A Right logical shift of one position moves each bit to right by one. The vacant least significant bit (LSB) is discarded and in place of MSB 0 is added

Left logical shift



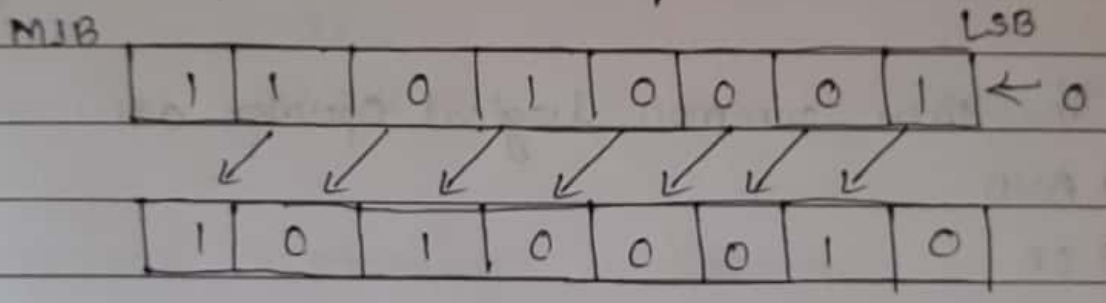
Right logical shift



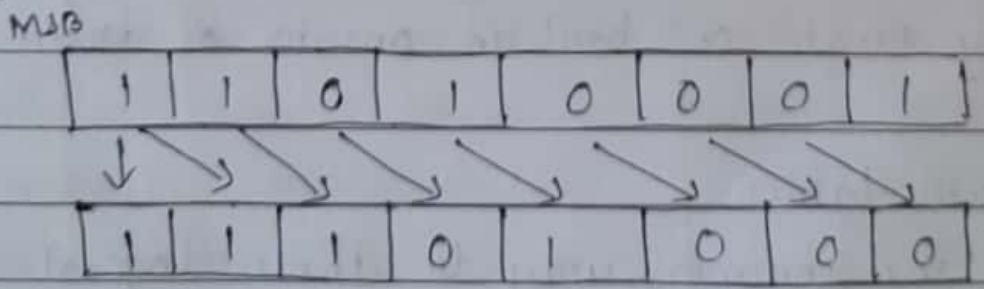
Arithmatic shift

A left arithmatic shift of one position moves each bit to left by one. The vacant least significant bit (LSB) is filled with zero and most significant bit is discarded or stored in carry forward.

Left arithmatic shift



Right arithmatic shift of one position moves each bit to the right by one. The least significant bit is discarded and the vacant MSB is filled with value of previous MSB



5 Arithmatic operations are used to perform mathematical calculations. logical operators are used to perform logical opeations and include OR, AND & Boolean operations include AND, OR, XOR & NOT and can have one of two values true or false

The arithmatic operations are Addition (+), Substraction (-), multiplication (*), division (/) & exponential (^)

The three common logical Operator are

* AND
* OR
* NOT

The logical operators are often used to create a test expression that controls program flow. This type of expression is also known as Boolean expression because they create a boolean answer or value when evaluated

Truth tables

The common way to show logical relationship is in truth table

x	y	x and y	x or y
false	false	false	false
false	true	false	true
true	false	false	true
true	true	true	true

x	not x
false	true
true	false

Operator	Meaning	Example	Result
&&	Logical AND	(6>4)&&(7>5)	false
||	Logical OR	(6>4) || (7>5)	True
!	Logical NOT	!(6>2)	True

7. Endianness is the order or sequence of bytes of a word of digital data in computer memory. Endianness is primarily expressed as
* Big Endian (BE)
* Little Endian (LE)

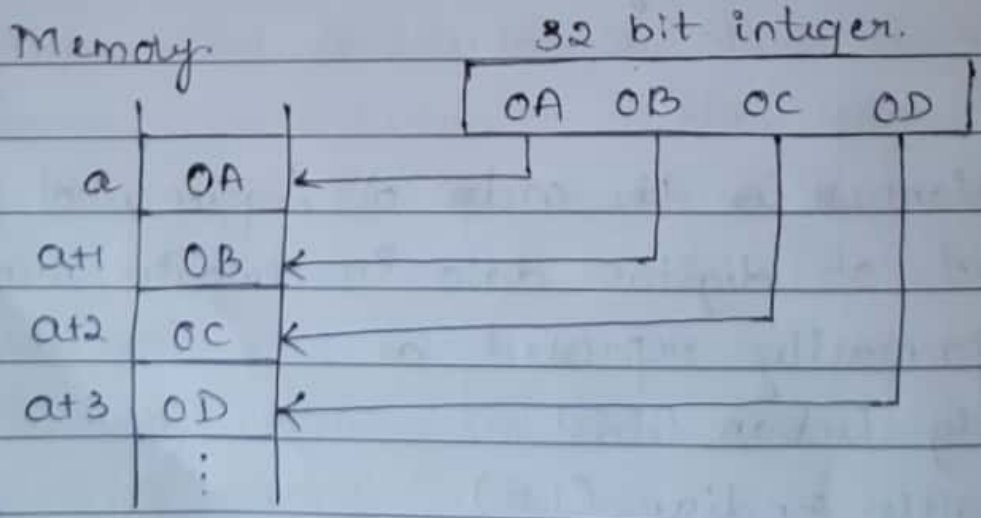
A big Endian system stores the most significant byte of a word at the smallest memory address and least significant byte of the word at the

largest memory address

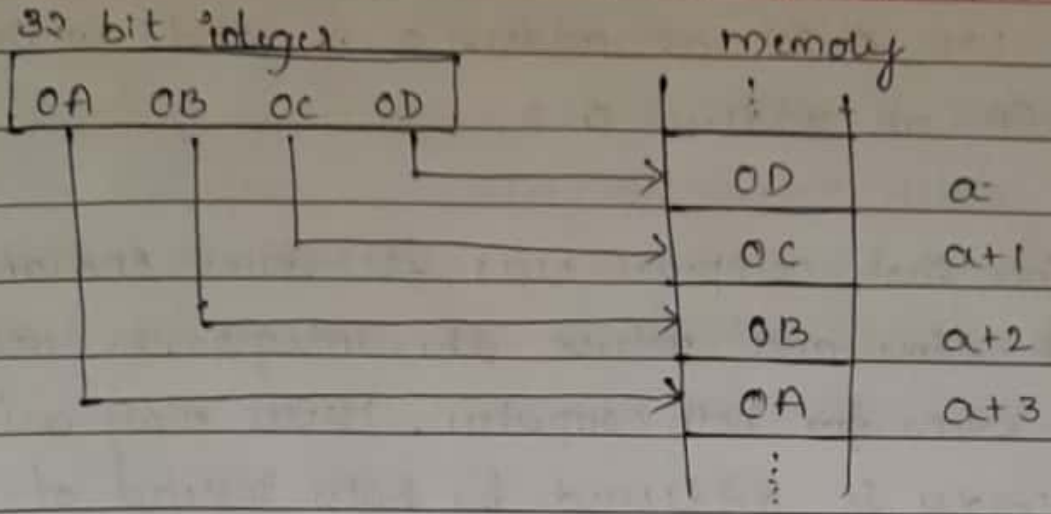
A little Endian system stores the least significant byte of the word at the smallest address and most significant byte of the word at the largest memory address

Endianness may also be used to describe the order in which the bits are transmitted over a communication channel.

For ex: Big Endian in a communications channel transmits the most significant bit first



(a) Big endian



(b) Little Endian

The two diagrams show how two computers use diffaunt endianness Store a 32 bit integer with the value of 0x0A0B0C0D

In both cases the integer is broken into 4 bytes 0x0A 0x0B 0x0C 0x0D and are stored in four sequential bytes location in memory starting with the memory location with address a, a+1, a+2, a+3 the difference

The diagram (a) shows a computer using Big Endian This starts the storing of the integer with MSB 0x0A at the address a & ends with LSB 0x0D at address a+3

The diagram (b) shows a computer using little endian. This starts the storing of the integer with

the LSB 0x0D at address a+3 ends with the MSB 0x0A at address a+3

Since each computer uses its same endianess to both store and retrive the integer the result will be same for both computers. Issues may arise when memory is addressed by bytes instead of integers when memory contents are transmitted between computers with different Endianness.

8 Virtualisation is technology that lets you create useful IT services using resources that are traditionally bound to hardware.

Types of Virtualisation

* Presentation Virtualisation

Its an application delivery method that delivers desktops or application from a shared server. This enables access to client application from a central server that is connected with clients. This initiate a presentation instance. The only resources shared with the client is the graphical user interface as well as the mouse/keyboard.

* Virtual desktop Infrastructure (VDI)

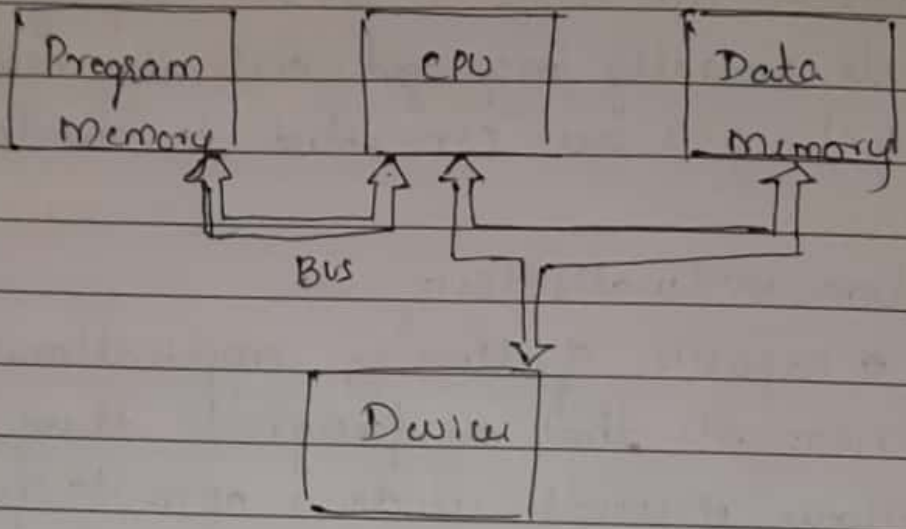
Sharing similarities with present virtualisation VDI Solutions are also a remote display protocol that hosts centrally managed virtual machines that client pc's are connected to one-to-one network

* Application virtualisation

Its a capable of allowing applications to run in environments that are foreign to them, For example Wine allows microsoft windows apps to run on Linux. By establishing a common software baseline across multiple computers with an organisation application. virtualisation also reduces system integration and administration cost. Finally it enables simplified operating system migrations, whereby application can be transferred to removable media or between computers without having to install them effectively becoming portable software.

1. Continue

Harvard Architecture



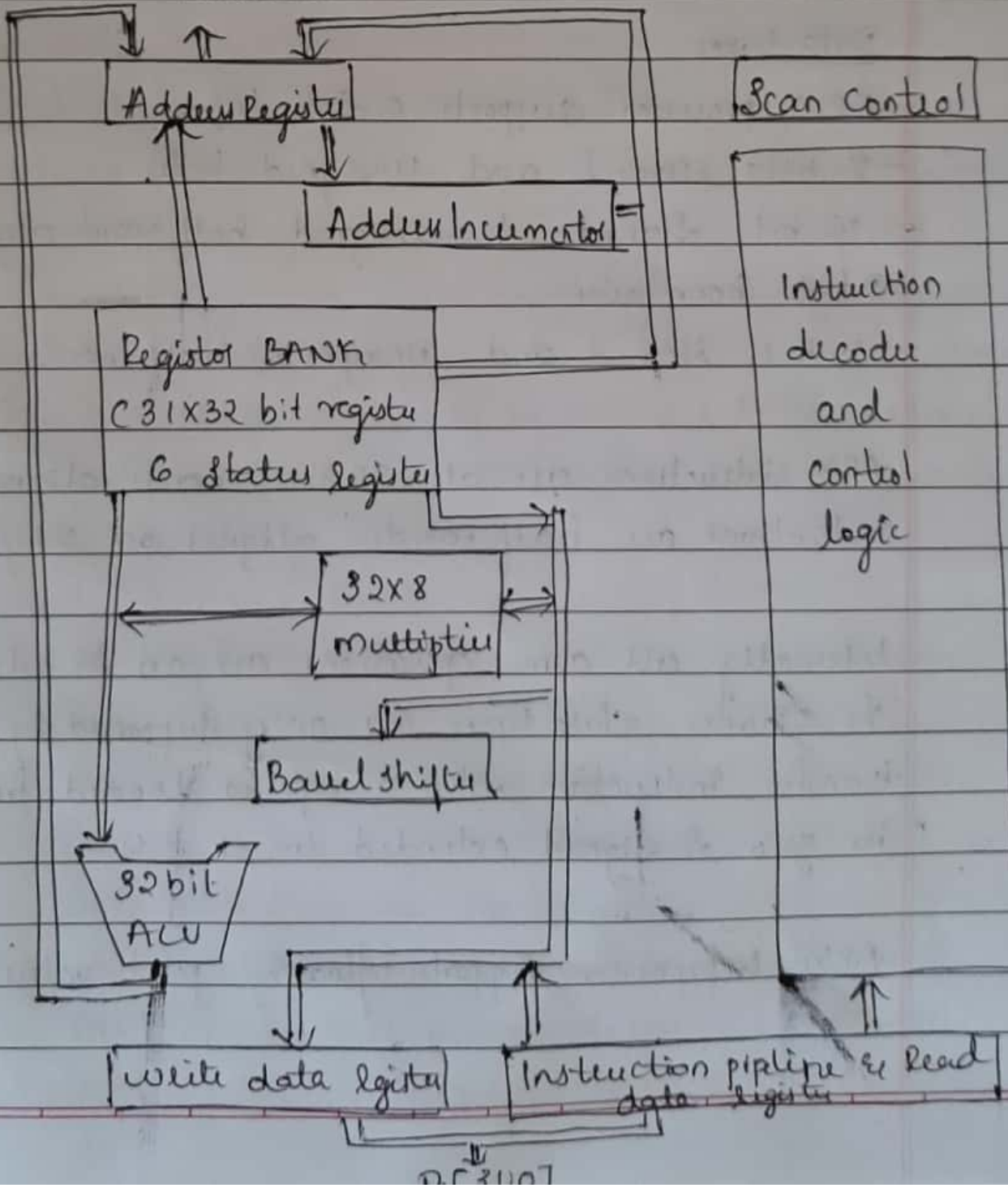
- memory from data is seperated from memory from instruction
- Since it has two memories, this allows parallel access to data & instructions.
- development of the control unit is expensive & needs more time
- data and instruction are accessed the same way
- Both memory can use different cell sizes

3. With a neat diagram explain ARM7 architecture Feature used

- Load/Store Architecture
- Fixed length 32 bit instructions
- 3 address instruction format

ARM process is a 32 bit architecture most ARMs implementation two instruction sets
- 32 bit ARM instruction set
- 16 bit Thumb instructions.

→ Von neuman Architecture
→ 3 stage pipeline - fetch, decode, execute
→ 32 bit databus.



- 32 bit Addrss Bus
- 37-32 bit Registers
- 32 bit ARM Instruction Set
- 16 bit Thumb instruction Set
- 32X8 multiplier
- Barrel shifter

Data types

ARM processor supports 6 data types.
- 8 bits' signed and unsigned bytes
- 16 bit signed & unsigned half word aligned on 2 byte boundaries
- 32 bit signed and unsigned half words

ARM instructions are all 32 bits words aligned thumb instructions are half words aligned on 2 byte boundries

Internally all arm operations are on 32 bit operands The shorter data types are only supported by data transfer instruction when a byte is loaded from memory its zero or signed extended to 32 bits

ARM coprocessor supports floating point values.

4. with a neat diagram explain the programming model of ARM

- A processor instruction set defines the operations that the programmer can use to change the state of the system incorporating the processor
- This state usually comprises the value of data items in the processors visible registers and system memory
- Each instruction can be viewed as programing a defined transformation from the state before the instruction is executed to the state after it has completed
- when coding user level programs, only the 15 general purpose 32 bit registers (r_0 to r_{14}), the program counter (r_{15}) & CPSR need to be considered. The remaining registers are used only for system level programing & for handling exceptions

Processor modes.

CPSR[4:0]	mode	use	Registers Register
10000	user	Normal user code	user
10001	FIQ	Processing fast interrupts	-fiq
10010	IRQ	Processing std interrupts	-irq
10011	SVC	Processing software interrupts	-svc
10111	Abort	processing memory faults	-abt
11011	undef	Handling undefined instruction	-und
11111	System	Running privileged OS	user

CPSR
r_0
r_1
r_2
r_3
r_4
r_5
r_6
r_7
r_8
r_9
r_{10}
r_{11}
r_{12}
r_{13}
r_{14}
r_{15}
CPSR

FIQ	IRQ	SVC	und	Abort
r_8				
r_9				
r_{10}				
r_{11}				
r_{12}				
r_{13}(SP)	r_{13}(SP)	r_{13}(SP)	r_{13}(SP)	r_{13}(SP)
r_{14}(LR)	r_{14}(LR)	r_{14}(LR)	r_{14}(LR)	r_{14}(LR)
SPSR	SPSR	SPSR	SPSR	SPSR

* most programmes operate in usr mode ARM has other privilages operating modes which are used to handle exceptions, supervisor calls & System mode

- mode access rights to memory systems and to processors
- current operating mode is designed by CPSR[4:0]

Supervisor mode

> having some protecting privilages
- System level function can be accessed through specified supervisor calls.
- usually implemented by software interrupt

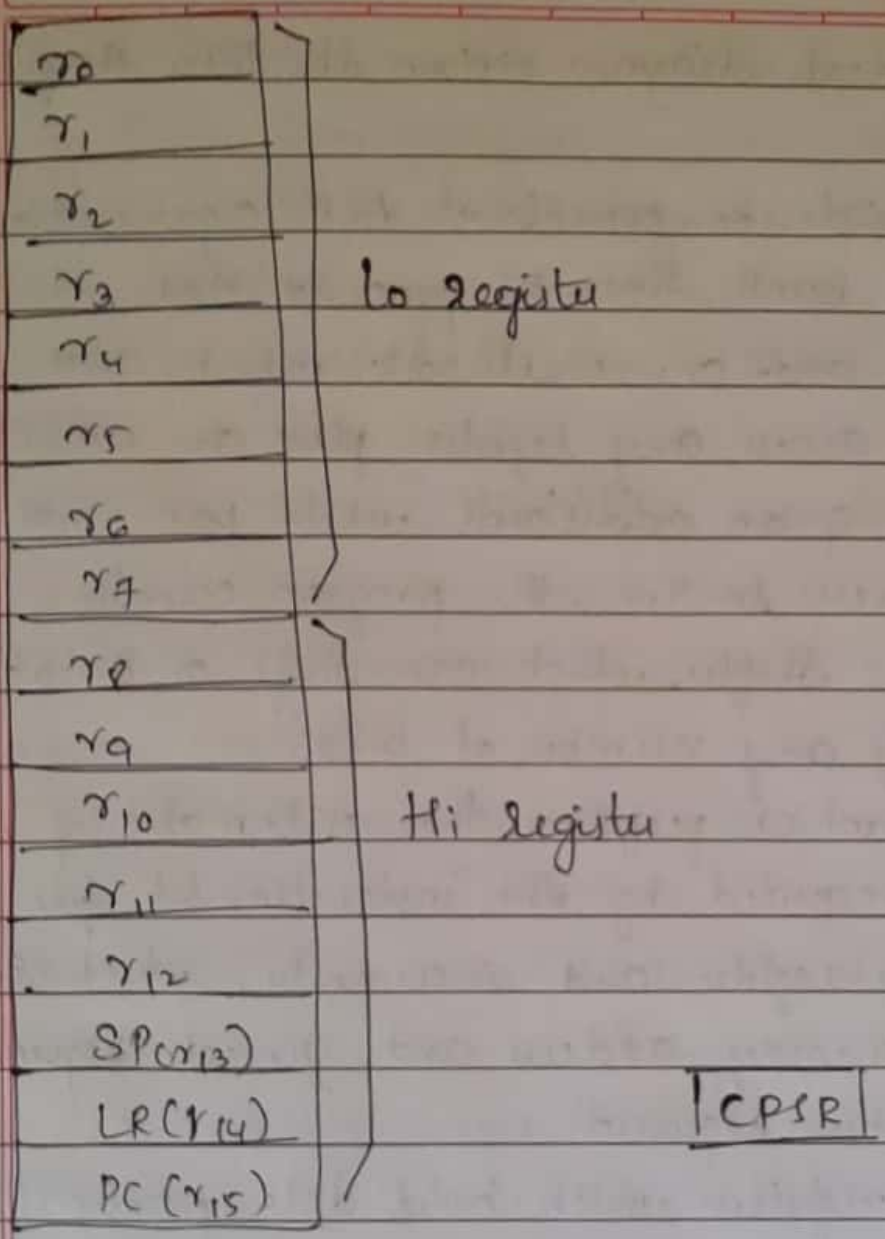
- Each mode can access
- a particulas set of R0-R5 registers
- Stack pointer, R14 - Link register
- program counter (R15)
- current program status register (CPSR)

5. with neat diagram explain the programing model of Thumb.

> The thumb instruction set is a subset of the ARM instruction set of instruction operate on a restricted view by of ARM register
> The instruction set gives full access to the eight 'Lo' general purpose register r0 to r7, & modes extensive use of r13 and r15 for special purpose
- r13 is used as stack pointer
- r14 is used as link register

- r_{15} is the program counter
- Their uses follow very closely the way that registers are used by the ARM instruction set. Though the use of r_{13} as a stack pointer in ARM code is purely a software convention whereas in thumb code it is somewhat hard coded. The remaining registers (Apart r_{12} and the CPSR) have only restricted access.
- A few instructions allow the 'hi' registers 'r_8 to r_{15}' to be specified.
- The CPSR code flags are set by arithmetic & logical operation and control conditional branching.

- All thumb instruction are 16 bit long. They map onto ARM instruction so they inherit many properties of the ARM instruction set
- The load-store architecture with data processing data-transfer & control flow instructions
- Support for 8 bit byte, 16-bit half word and 32 bit word data types where 3 half words are aligned on 2 byte boundaries and word are alligned on 4-byte boundaries
- A 32 bit unsegmented memory



→ However inorder to achieve a 16 bit instruction length a number of characteristic features of the ARM instruction set have been abandoned

- most thumb instruction are executed unconditionally
- most thumb data processing instruction use a 2-address format c the destination register is same as one of the source registers.
- Thumb instruction formats are less regular than ARM

Instruction format are less regular than ARM instruction format as a result of dense encoding

Page No.: RELIABLE
Date : / /

6. With a neat diagram. Explain the three stage pipeline of ARM.

→ The principle components of ARM organisation are

- The register bank, stores the processor state. It has two read ports and one write port which can each be used to access any register plus an additional read port & an additional write port that give special access to r15, the program counter.
- The barrel shifter, which can shift or rotate one operand by any number of bits
- The ALU which performs the arithmatic & logic function required by the instruction set. ~~function~~
- The address register and incrementer, which select & hold all memory address and generate sequential address when required.
- The data register which hold data going to and from memory
- The data register which hold data
- The Instruction decoder & associated control logic

The 3 stage pipeline:

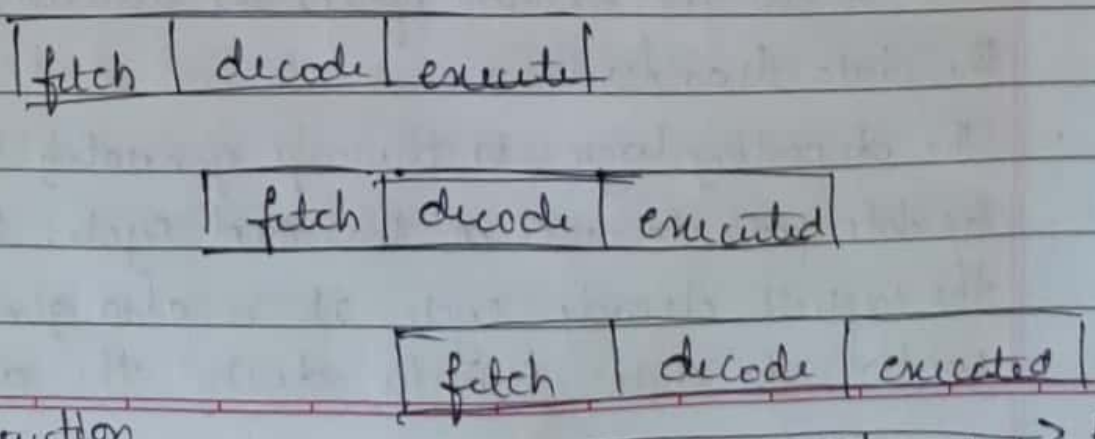
ARM processor upto ARM7 employ a simple 3 stage pipeline with the following pipeline stages.

Page No.: RELIABLE
Date: / /

Fetch: The Instruction is fetched from memory & placed in Instruction pipeline.

- decode: The Instruction is decoded and the data path control signals prepared for the next cycle. In this stage the instruction 'owns' the decode logic but not data path.
- Execute: The Instructions own a data path. The register bank is read, an operand shifted, the ALU result generated & written back into a destination register.
- At any point of time, three different instruction may occupy each of these stages so the hardware in each of these stages so the hardware in each stage has to be capable of independent operation.

When the processor is executing simple data processing instruction the pipeline enables one instruction to be completed every clock cycle. An individual instruction takes 3 clock cycle latency, but the throughput is one instruction per cycle



- when multi cycle instruction is executed the flow is less regular

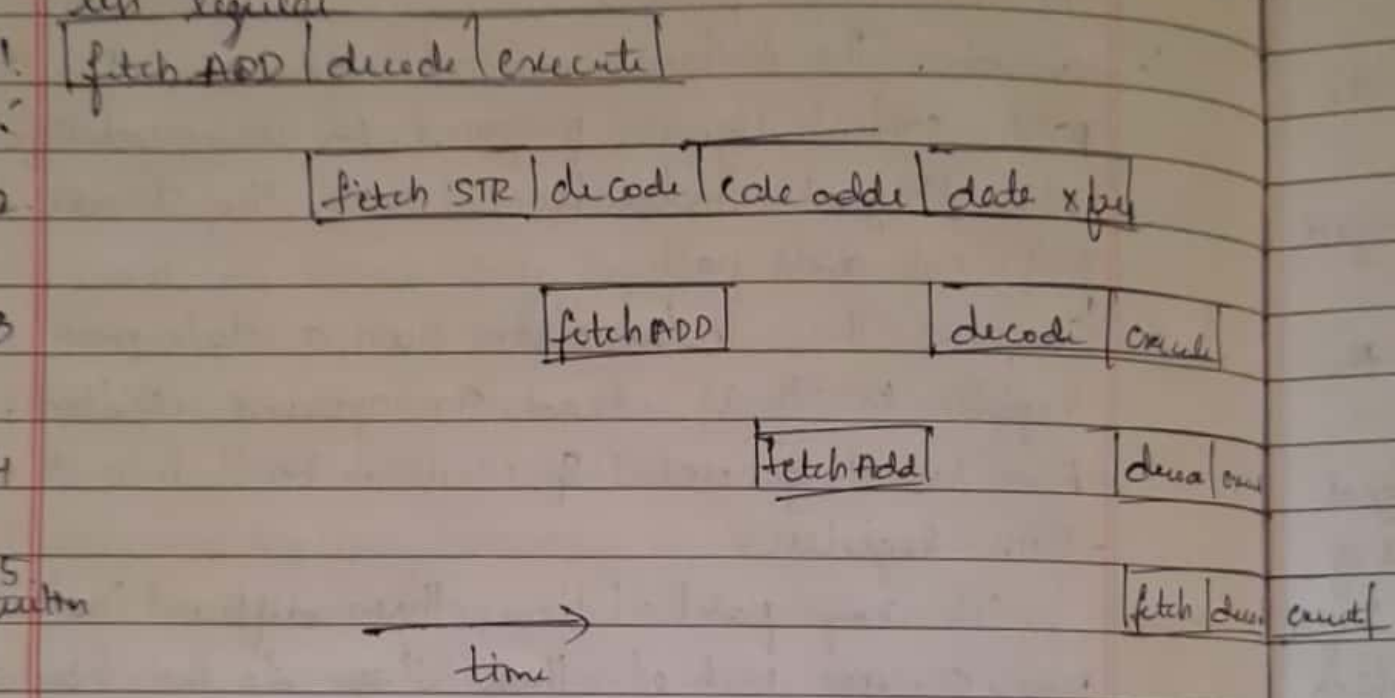


fig ARM multicycle instruction 3 Stage pipeline operation

- The figure shows a sequence of single cycle ADD instruction with a data store instruction, STR occuring after the first ADD, The cycles that access main memory are used in every cycle.
- The datapath is likewise used in every cycle, being involved in all the execute cycles the address calculations & the data transfer.
- The decoding logic is always generating the control signals for data path to use in the next cycle. So in addition to the explicit decode cycle it is also generating the control for data transfer during the address calculation

Cycle of STR

Thus in this instruction sequence, all parts of the process are active in every cycle & the memory limiting factor, defining the number of cycles the sequence must take

7 With a neat diagram, explain CPSR register.

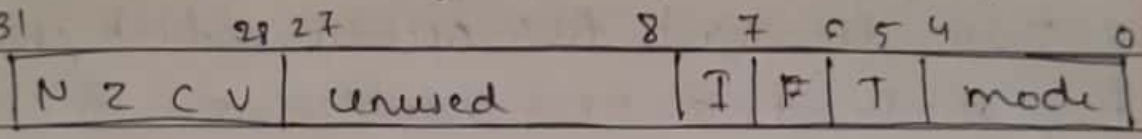


Fig: ARM CPSR format

cont.

- The CPSR is used in user level program to store the condition code bits - These bits are used for ex, to record the result of a comparision operation & to control whether or not a conditional branch is taken
- The bits at the bottom of register control the processor mode

CPSR [4:0]	mode	use	Register
10000	User	Normal user code	user
10001	FIQ	Processing fast interrupt	-fiq
10010	IRQ	Processing standard interrupt	-irq
10011	SVC	processing software interrupt	-svc
10111	Abort	processing memory fault	-abt
11011	Undef	Handling undefined instruction trap	-und
11111	System	Running privileged os tasks	user

the Instruction Set & T interrupt enable 'I' & 'F' are protected by user level program.

- The condition code flags are in top four bits of the register & have the following meanings.

→ N: Negative :- The last ALU operation which changed the flag produced a negative result.

→ Z: Zero: the last ALU operation which changed the flags produced a zero result.

→ C: Carry : The last ALU operation which changed flags generated a carry out either as a result of an arithmatic operation in the ALU or from the shifter.

- V: Overflow: The last arithmatic ALU operation which changed the flags generated an overflow into the sign bit.

Q) Explain the seven different modes of ARM

→ Mode changes can be made under software control or can be caused by external interrupts or exception processing.

Most application programs ~~can~~ be executed in user mode. while process is in user mode, the program being executed is unable to access some protected system resources or to change mode, other than by causing an exception to occur. This allows a suitably written OS to control the use of system resources.

The modes other than user mode are known as privileged modes. They have full access to system resources & can change mode freely. Five of them are known as exception modes: FIQ (Fast Interrupt), IRQ (Interrupt) Supervisor, Abort & undefined, IRQ (Interrupt) These are entered when specific exception occurs

The remaining mode is System mode, Its not entered by any exceptions & has exactly the same registers available as user mode. However its a privileged mode & is therefore not subject to the user mode restrictions

User mode is the usual arm program execution State and is used for executing most application programs.

- Fast interrupt (FIQ) mode supports a data transfer & channel process.
- Interrupt mode is used for general purpose interrupt handling.
- Supervisor mode is protected mode for the OS.
- Abort mode is entered after data & instruction prefetch abort
- System mode is privileged user mode for OS.
- We can only enter System mode from another privileged mode by modifying the mode bit of CPSR
- Undefined mode is entered when an undefined instruction is executed

- modes other than user mode are collectively known as privileged mode. Privileged modes are used to service interrupts or exceptions or to access protected resources.

9. Explain the nomenclature in ARM

ARM was originally from Acron Computer Ltd. First RISC processor for commercial use ARM 7 TDMI processor.

32 bit processor advanced Machine.

-T -> Thumb Architecture Extension

D → Debug extension.

M → Enhanced Extension

I → Incircuit Emulation.

ARM {x} {y} {z} TDMI {E} {J} {F} {S}.

x → Series S → Synthesizable version.

y → memory management unit

z → cache

T → Thumb 16 bit decoder.

D → JTag debugger

M → Fast multiplier

I → Embedded ICE

E → Enhanced Instruction for DSP

J → Java acceleration by Jazelle.

F → Floating point

RELIABLE
Page No.:
Date : / /

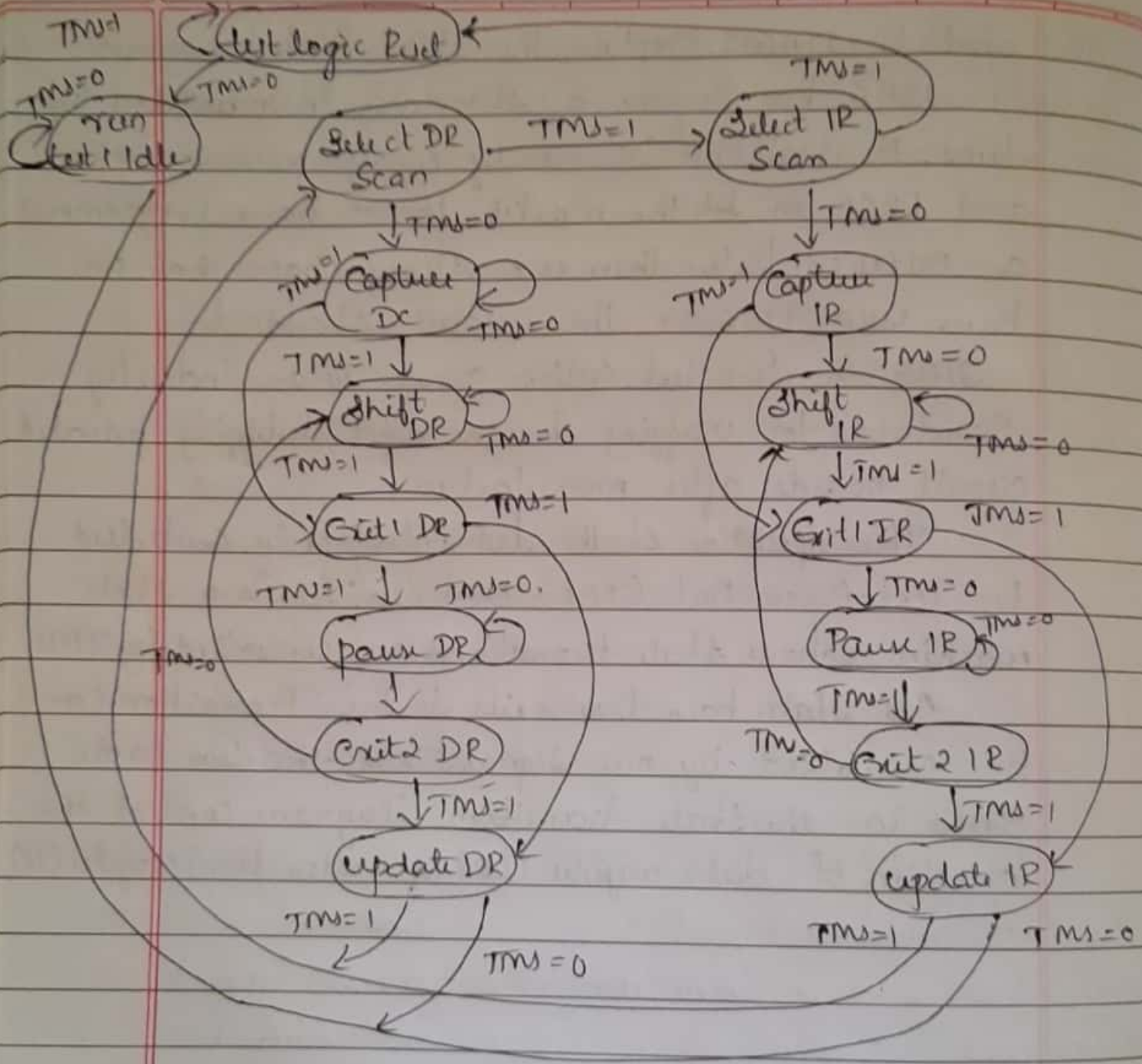
10. What is JTAG? Explain the JTAG State diagram.

JTAG has become a standard in embedded systems. Its available in nearly every microcontroller and FPGA on ~~the~~ the market. If we have programmed a microcontroller there is a strong chance that we have used JTAG or the related standard.

JTAG is Joint test action group. is an Industry standard for verifying designs and testifying printed circuit boards after manufacture.

The operation of the test interface is controlled by Test Access Port (TAP) controller. This is a state machine whose state transitions are controlled by TMS.

All states have two exits so the Transition can be controlled by one signal, TMS the two main paths in the state transition diagram control the operation of data register (DR) & instruction register (IR)



The normal operation of JTAG test system is to enter an instruction which specifies the sort of test to be carried out next & the data register to be used for the test into the instruction register & then to the use the data register to carry out the test

Instruction may be public or private. Public Instruction are declared and available for general test use, & the Standard Specifies a minimum set of public Instruction that must be supported by all devices that comply with the Standard. Private Instruction for the Specialized on chip test purposes and the Standard doesn't specify how these should operate or how they should be used.

The interface works with 5 dedicated signals which must be provided on each chip that supports the test Standard

- $\overline{TRST}$ is a test reset input which initializes test interface
- TCK is test clock which controls the timing of the test interface independent from any test clock.
- TMS is the test mode Select which controls the operations of the test interface State machine.
- TDI is the test data input line which carries the sampled values from the boundary Scan chain & propagates data to the next chip in the Serial test circuit

RELIABLE
Page No.:
Date: / /

11. What is single tasking. Give example of processor with MMU support

Single tasking means doing one task at a time with as little distraction & interruption as possible

Microcontroller are known as computer on chip. They are designed to perform a single task only because its processing power as well as memory is not suitable for installing an OS

12 What is MMU? Why MMU is required? Give example of MMU support

→ The memory can be defined as a collection of data in a specific format. It's used to store instruction and processed data. The memory comprises a large array of group of words or byte. Each with its own location The primary motive of a computer system is to execute programs. These programs along with information by access should be in the main memory during execution. The CPU fetches the instruction from memory according to the value of PC

The main memory is central to the operation of a computer. Main memory is a large array of words and bytes ranging in size from hundreds to thousands to billions. Main memory is a repository of rapidly available info shared by CPU & IO devices

main memory is the place where programs and info are kept when the processor is effectively utilizing them. Main memory is associated with the processor is extremely fast. Main memory is also known as RAM, which is volatile memory. RAM eats its data when power interrupt -ion occurs.

Memory Management.

In a multiprogramming computer the OS resides in a part of memory & kernel kernel is used by multiple processes. The task of subdividing the memory among diffrent process is called as memory management. Memory management is a method in OS to manage operation between main memory and disk during process execution The main aim of memory management is to achieve efficient utilization of memory.

memory management is required to

- allocate and deallocate the memory before and after process execution
- To keep track of used memory space by processes.
- To minimize fragmentation issues.
- For proper utilization of main memory
- To maintain data integrity while executing of process.

Ex: IBM System / 360 model 67, IBM System 370.

RELIABLE
Page No.:
Date: / /

ARM:
ARM architecture based applications processors implement an mmu defined by ARM's virtual memory system architecture. The current architecture defines PTE's for describing 4KB and 64 KB pages. 1MB sections & 16 MB super section legacy versions also defined 1KB tiny space

14) Write a c program to find the endianness of a given number.

⇒ Program 1:

```
#include<stdio.h>
int main(void)
{
 unsigned int value = 0x11223344;
 char *r = (char *) &value;
 int i;
 for (i=0; i<4; i++)
 {
 printf("Addres of 0x%.x = %d \n", r[i], &r[i]
 }
 return 0;
}
```

program :

```
#include <stdio.h>
int main(void)
{
  unsigned int value = 0x1;
char *r = (char *) &value;
if (*r == 1)
  printf ("your system is little Endian\n");
else
printf ("your system is Big Endian\n");
return 0;
}
```

15 Explain bit, byte, Nibble, Half word, word

→ Binary values are often groupped into a common length of 1's & 0's. This number of digits is called the length of a number. Common bit lengths of binary numbers includes. bits, nibble, bytes, Each 1 or 0 in a binary number is called a bit

- A group of 4 bits is called a nibble.
- A group of 8 bits make a byte

length	name	example
1	Bit	0
4	nibble	1011
8	word Byte	1011 1010
16	Haft word	1011101010111010

word is another length that gets thrown out from time to time. Word is much less standing & more ambigious. The length of a word is usually dependent on a architecture of a processor It could be 16 bits, 32 bits, 64 or even more

The terms half word or single word are often used in contemporary computing to refer to common word size relative to a 32 bit base word size

half word = 16 bits

word = 32 bits.

16. Explain the word align & halfword align in ARM memory.

→ Different processes have different defination of words. For 32 bit processors a word is 32 bit (4 byte) as the name implies, a half word is 16 bit for a 16 bit processor, a word is 16 bits (2 bytes), for 8 bit processor word is 8 bits

word alignment: The stored address are adjacent & can be divided by 4, the last two digits are 00.
Half word alignment: The stored address are adjacent & can be divided by 2. that is the last bit is 0.
ARM instructions require 32 bit ARM instructions that must be word aligned & stored in memory & 16 bit thumb instruction required half word aligned and stored. Therefore in ARM state the value of R15 is always divisible by 4, that is lowest 2 bits of R15 register are always 00. In the thumb state, the value of R15 is always divisible by 2, which is the lowest bit of R15 register always 0. One word consists of one or more bytes. ie usually integer bits of bytes.

17 Explain the software tools involved in processing the C source file with neat diagram.

→ Software development for ARM is supported by a coherent range of tools development by ARM & there are many third party & public domain tools available such as an ARM backend for the gcc C compiler.

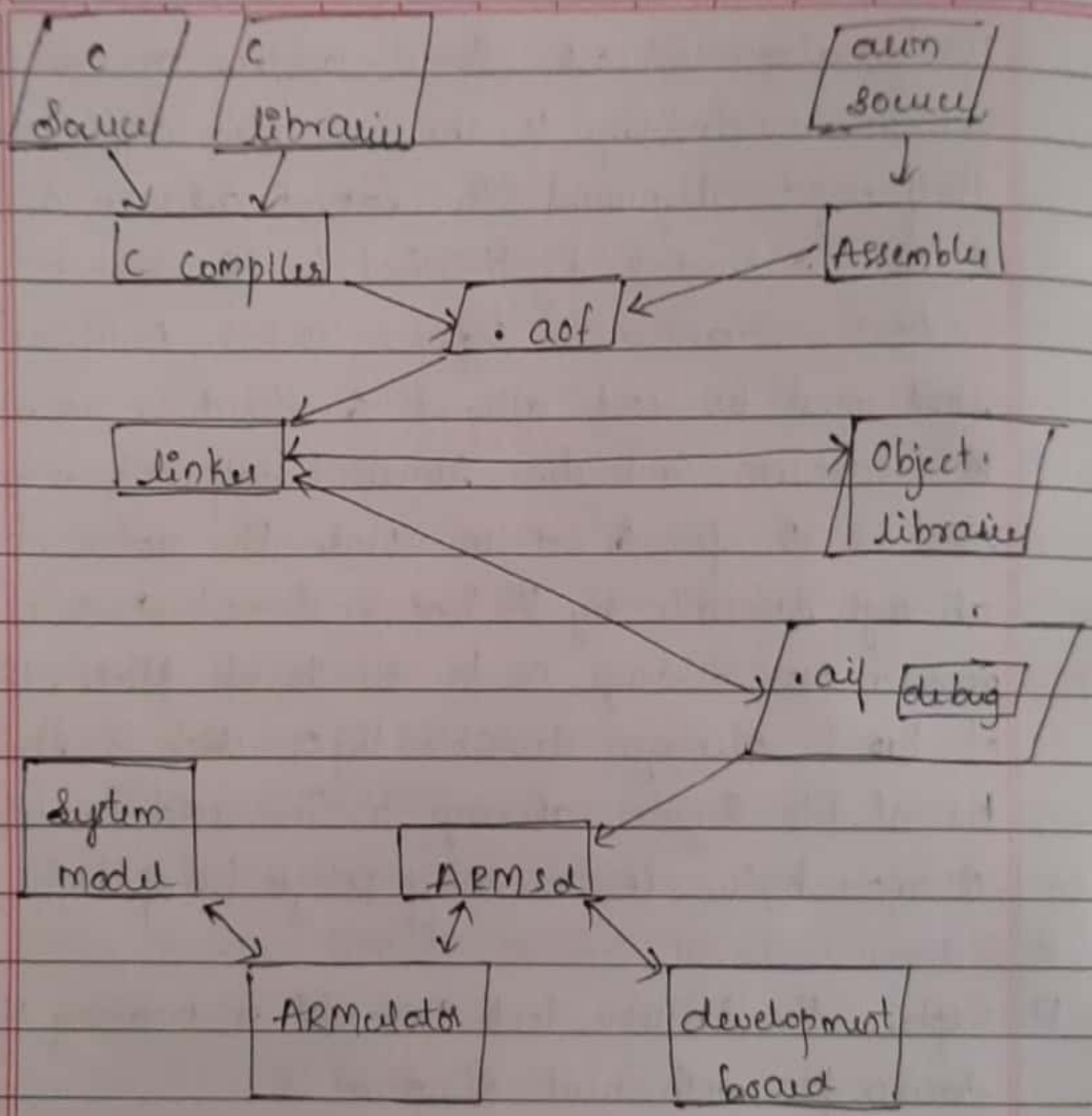


Fig: Structure of ARM cross-development Toolkit

ARM C compiler: It uses the ARM procedure call standard for all externally available functions

ARM assembler: It's a full macro assembler which produces ARM object format output that can be linked with output from C compiler.

The Linker: The linker takes one or more object files & combines them into an executable program.

ARMsd: The ARM symbolic debugger is a front end interface to assist in debugging programs running either under emulator or remotely on a target system such as ARM development board.

ARMulator: The ARM emulator is a set of programs that models the behaviour of various ARM processor cores in software on a host system.

18) Explain the following addressing modes in ARM
a) Three address
b) Two address
c) Single Address instruction with respect to ARM

→ Sequence of instructions forms a program to perform a specific tasks.

2 components → Opcode field → Specifies how data manipulated
→ Address field → Specifies the data location.

When data to be read from or stored into, two or more address field may have one or more than one address.

* Three address Instructions
* Two address Instructions.
* one address Instructions
* Zero address Instructions.

processor can execute an instruction only if it is represented in Binary sequence.

A unique binary sequence pattern must be assigned This process is called op-code encoding.

One address Instruction

This uses an implied accumulator register for data manipulation and the other is in the register or memory location. Implied means that the CPU already knows that one operand is in the accumulator so there is no need to specify it.

Ex: LDR addr.

ACC ← (addr)

Two address Instructions:

~~There is~~ three two address can be specified in the instruction. In the one address instruction the result was stored in the accumulator., here the result can be stored in different location. i.e, register or memory location. But requires more number of bit to represent the address.

Ex: MOV R1, R2

$R_1 \leftarrow [R_2]$

Three address Instructions:

This has three address field to specify a register or memory location. Program created are much shorter in size. but number of bits per instruction increase. Program

created are much short in size but number of bits per instruction increase. These instructions make creation of programs much easier but it doesn't mean that program will run much faster because how instruction only contain more info but each micro operation will contain more to be performed in one cycle only. Ex: ADD R3, R1, R2 R3 = R1 + R2

R3

19. Explain the LPC 2148 Mc with block diagram.

→ Features

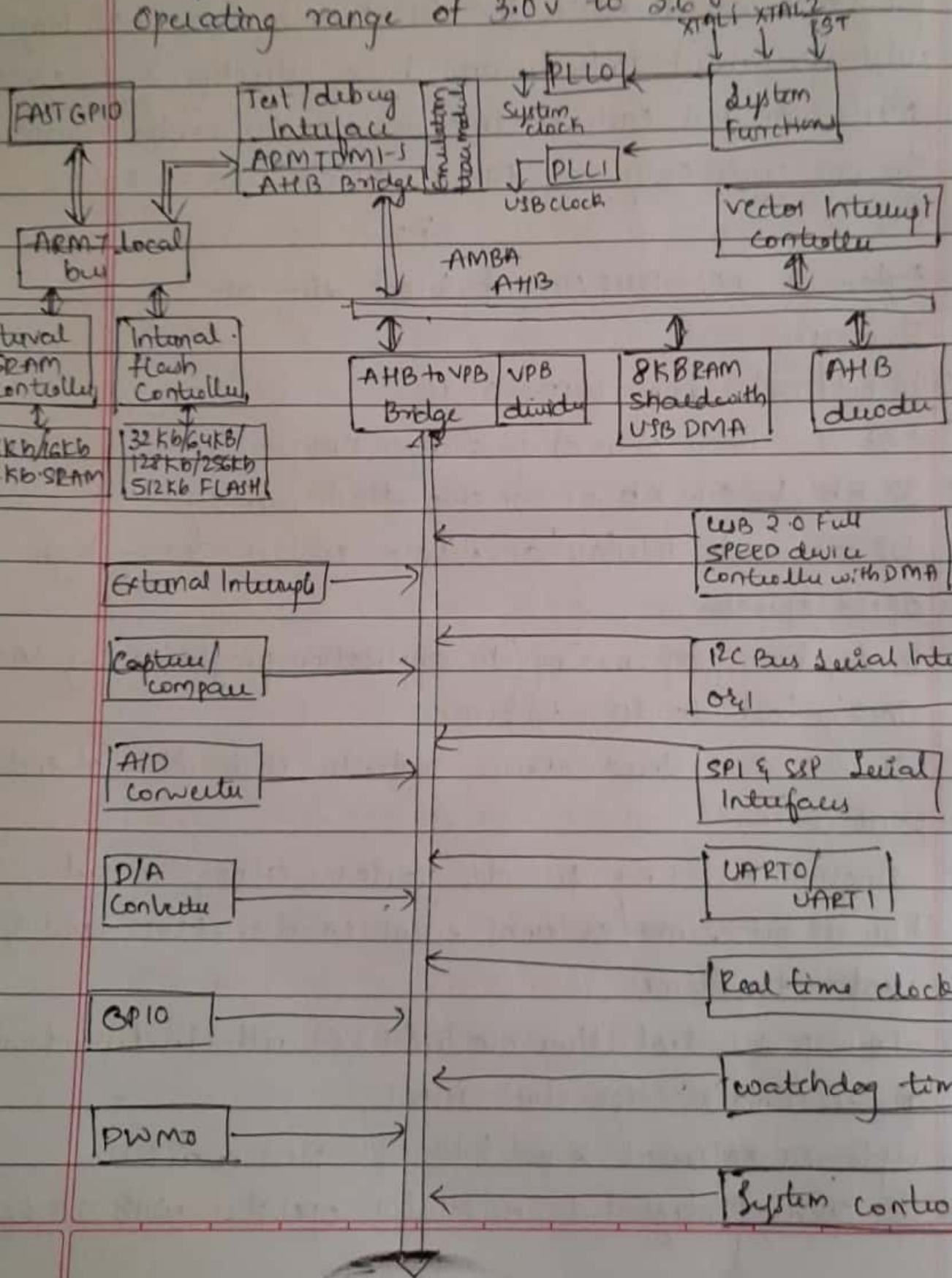
> 16 bit/32 bit ARM7TDMI-S Mc
> 8KB to 40KB of on chip static RAM.
> 32 Kb to 512 Kb of on chip flash memory
> 128 bit code interface/accelerator enables 60MHz high speed operations.
> In system programming / In application programming via on chip boot loader software.
> USB 2.0 Full Speed device controller with 2Kb of end point RAM.
> Single 10 bit DAC provides variable along output.
> Two 32 bit timer/external event counter, PWM unit & watch dog timer.
> Low power real time clock (RTC) with independent power and 32kHz clock input.
> upto 21 external inputs Interrupt pins available.
> The on chip integrated oscillator operates with an external

Page No.:
Date: / /

RELIABLE

Crystal from 1MHz to 25 MHz

> Single power supply with POR, BOD circuit CPU operating range of 3.0 V to 3.6 V



20. Explain the LPC 2148 µC GPIO pins.

→ GPIO – general purpose Input / output.

A 32 bit register used to select the function of the pin in which the user needs it to operate. There are four functions for each pin of the controller, which the first function is GPIO. It means that the pins can either act as an input or output with no specific functions.

There is totally 3 PIN Register in LPC2148 Controller in order to control the functions of the pins in the respective ports. The classification is given below.

PINSEL0 – Controls functions of PORT 0.0 – PORT 0.15

PINSEL1 – Controls functions of PORT 0.16 – PORT 0.31

PINSEL2 – Controls functions of PORT 1.16 – PORT 0.31

where the 00 of 32 bit register are GPIO configuration

00	GPIO Port 0.0
01	TXD (UART)
10	PWM1
11	Reserved

LPC 2148 has 2 32-bit GPIO port. A total of 30 I/O & a single output only pinout of 32 pins are available on Port 0. PORT 1 has upto 16 pins available for GPIO functions. PORT 0 & PORT 1 are controlled via 4 group of 4 registers.

* IOPIN * IOSET * IODIR * IOCLR

IOPIN: The register provides the value of port pin that are configured to perform only digital functions. The register will give the logic value of PIN. Regardless of whether the PIN is configured for input or output or as GPIO it can alternate digital functions.

IOSET: This register is used to produce a HIGH level output at port pins configured as GPIO. in an output mode. Writing 1 produces a HIGH level at the corresponding port pins. writing 0 has no effect. If any pin is configured as an input or a decondary function. writing 1 to the corresponding bit in the IOSET has no effect.

IODIR: This word accessible reg is used to control the pins when they are configured as GPIO port pins. Direction bit for any pin must be set according to the pin functionality.

IOCLR: This register is used to produce a low-level output at port pins configured as GPIO in an o/p mode. writing 1 produces a low level at the corresponding port pins & clear the corresponding bit in IOSET register. writing 0 has no effect. If any PIN is configured as an input. or a decondary function writing to IOCLR has no effect.

Page No.: RELIABLE
Date : / /

21 Explain the LPC2148 with a neat diagram Explain Baud rates & Bit Rate. Explain the calculations.

→ Baud : How many time a signal change per second

Bit :- How many bits can be sent per time unit (usually per second)

Bit rate is controlled by baud & number of signal levels.

Baud rate:

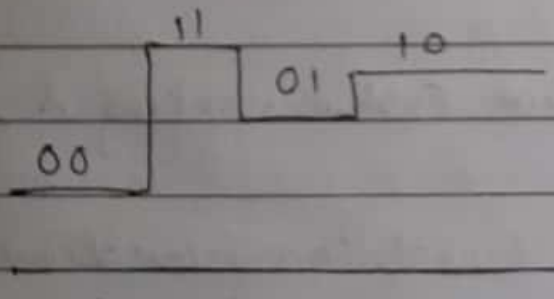
→ Number of time Unit changed per second

→ Let Baud rate be 4 (4 changes / per second)

→ Let bits per line be 2

→ Bit rate = 8 bits per second

→ Bit rate = 2 × Baud rate in this example



→ Baud rate defines the switching of signal

→ Bit rate defines the rate at which info flows across a data line measured in bits / second

An analog signal carries 4 bits in each signal unit If 1000 signal units are sent per second, find the Baud rate & the bit rate

1 Bit → 1 Symbol

Bit rate = Baud rate

1000

1000 = Baud rate

bauds/sec

Bit rate = 1000 x 4 = 4000 bps.

if Bit rate (or data rate) is "b"

Baud rate (or Symbol rate) is "s"

$b = s \times n$

b → data rate (bits per second)

s → Symbol ~~data~~ rate (Symbol/sec)

n → no of bits per second

if n = 1, Baud rate = Bit rate

n = 4, Bit rate = 4 x Baud rate

12. With a neat diagram. Explain working & features of SPI protocol

- SPI is synchronous serial communication interface specification used for short distance communication
- It was developed by motorola.
- SPI devices communicate in full duplex using a master slave - architecture usually with single master
- The master device originates the frame for reading and writing.
- SPI is called a four wire serial Bus, contrasting with three, two or one wired serial Bus.

Page No.: RELIABLE
Date: / /

. The SPI may be accurately described as a Synchronous Serial Interface.

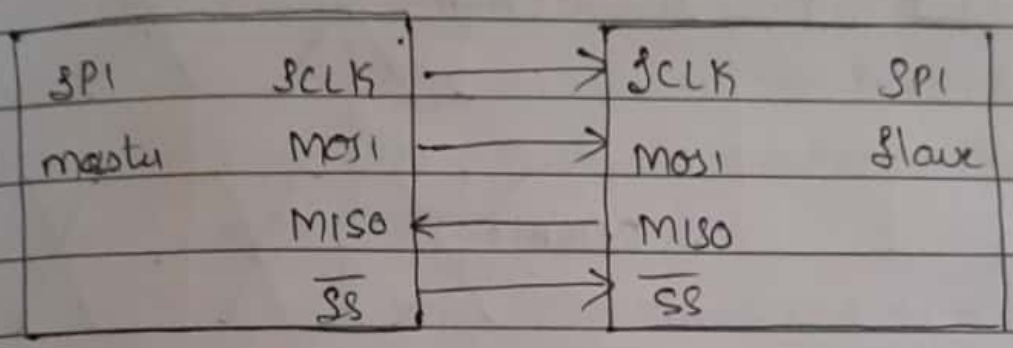


Fig: Single Master to Single Slave.

23 In SPI with a neat diagram explain CPHA & CPOL usage

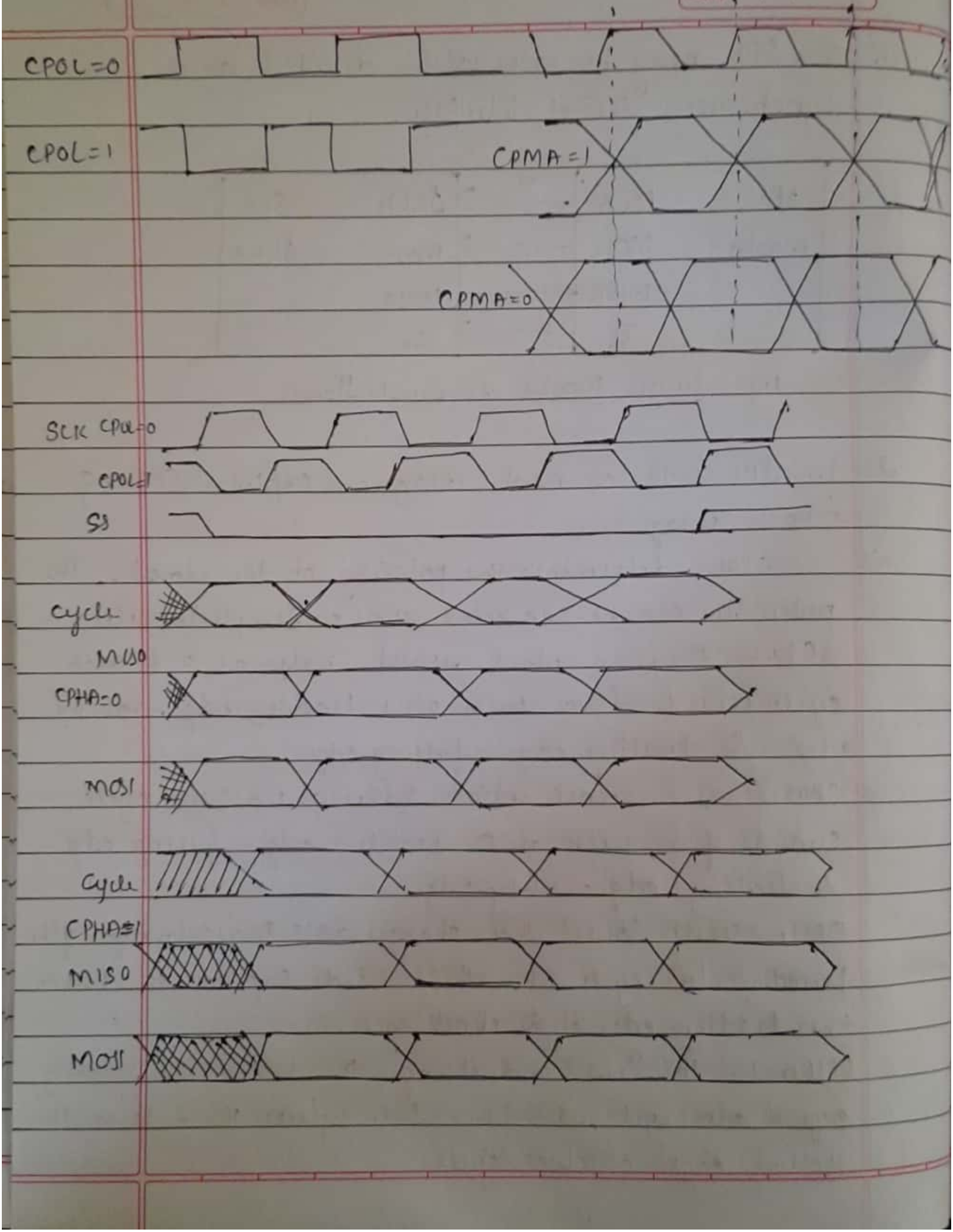
→ CPOL determines the polarity of the clock. The polarities can be converted with a simple inverter

CPOL=0 is a clock which idles at 0, & each cycle consists of a pulse of 1. leading edge - raising edge & trailing edge - falling edge

CPOL=1 is a clock which idles at 1 & each cycle consists of a pulse of 0. leading edge falling edge & trailing edge - raising edge

CPHA=0 the "out" side changes data on trailing edge of the preceding clock cycle, while the "in" side captures the data on the leading edge of the clock cycle.

CPHA=1 the "out" side changes the data on the leading edge of clock cycle, while "in" side captures the data on the trailing edge of clock cycle.

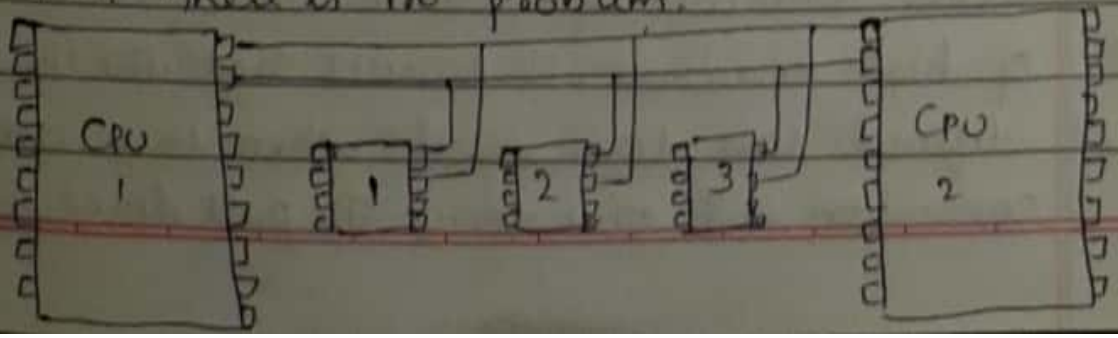


Page No.:
Date: / /

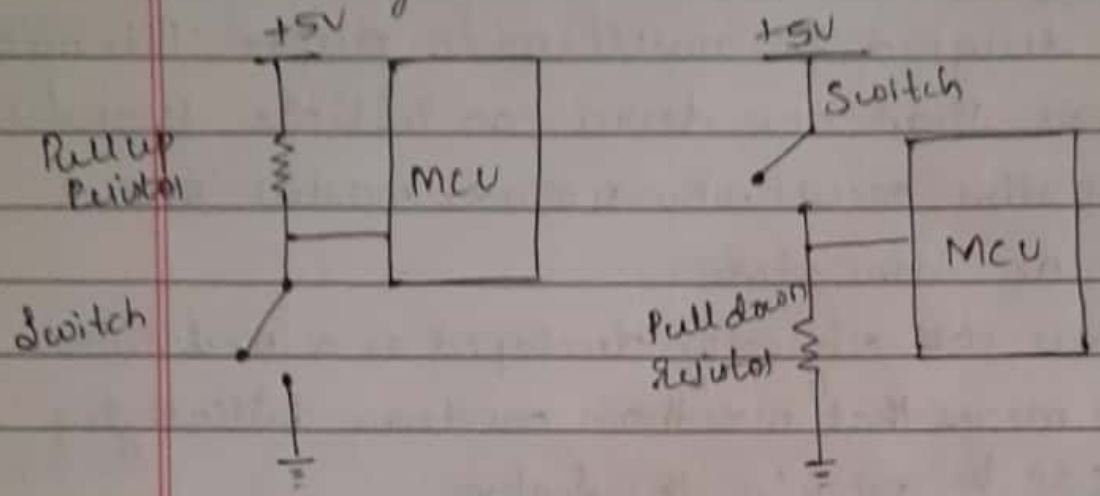
RELIABLE

16 with a neat diagram Explain the pull-up & pull down registers. Concept of arbitration in I2C

- I2C is designed for multi-master purpose. This means that more than one device can initiate transfers.
- Bus arbitration occurs when 2 or more masters start a transfer at same time.
- The I2C bus was originally developed as a multimaster bus this means that more than one device initializing transfer can be active in the system.
- When using only one master on the bus there is no risk of captured data. except if a slave device is malfunctioning or if there is a fault condition involving in the SDA / SCL bus.
- When MCU issues a start condition & sends the address. all slave will listen If the address doesn't match the address of CPU this device has to hold back any activity until the bus becomes idle again after stop condition.
- As long as two MCU's monitor what is going on the bus and as long as they are aware that a transaction is going on because that last issued command was not STOP there is no problem.



25 with a neat diagram Explain the pull up & pull down Resistor



Pull up Resistor :-

Its used to establish an additional loop over the critical components while making sure that voltage is well defined even when the switch is open

Its used to ensure that a wire is pulled to a high logic level in the absence of an input signal pull up Resistor which with a fixed values was used to connect the voltage supply & a particular pin in the digital circuit

Pull down Resistor :

A pull down Resistor is used to ensure that input that to the logic system settle at expected logic level whenever the external device are disconnected or of high impedence. It ensures that the wire is at a defined low logic level when there are no active connection with other device. The pull down Resistor holds

the logic signal near to zero volts when no other active device is connected.

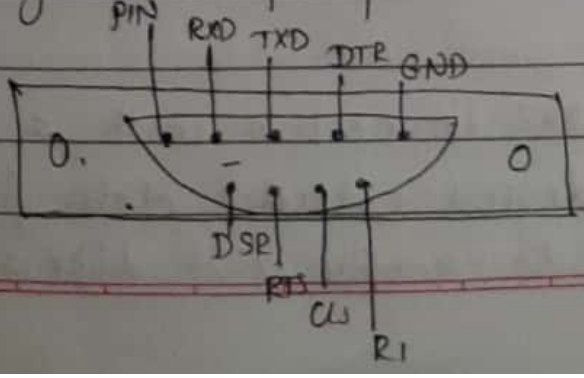
21 what is clock stretching. Clock Stretching in I2C

- It allows an I2C slave device to force the master device into a wait state. A slave device may perform clock stretching when it needs more time to manage data such as store received data or prepare the transmit another byte of data.

clock stretching In I2C devices can slow down communication by stretching SCL. During an SCL low phase any I2C device on the bus may additionally hold down SCL to prevent it to rise again enabling them to slow down the SCL clock rate or to stop I2C communication for a while. This is also refered to as clock synchro nization

In An I2C communication the master device determi nes the clock speed which relieve master and slave from synchronisation exactly to a predefined Baud rate

28. Explain working of DB9 pins & handshaking with the modem



Pin	Signal	Signal Name	DTE Signal Direction
1	DCD	Data carrier detect	In
2	RXD	Receive data	In
3	TXD	Transmit data	Out
4	DTR	data terminal ready	Out
5	GND	Ground	-
6	DSR	Data set ready	In
7	RTS	Request to Send	Out
8	CTS	clear to Send	In
9	RI	Ring Indicator	In

Handshaking modem:

A modem handshake is what occurs when the receiving modem answers the phone call & two modems begin to communicate

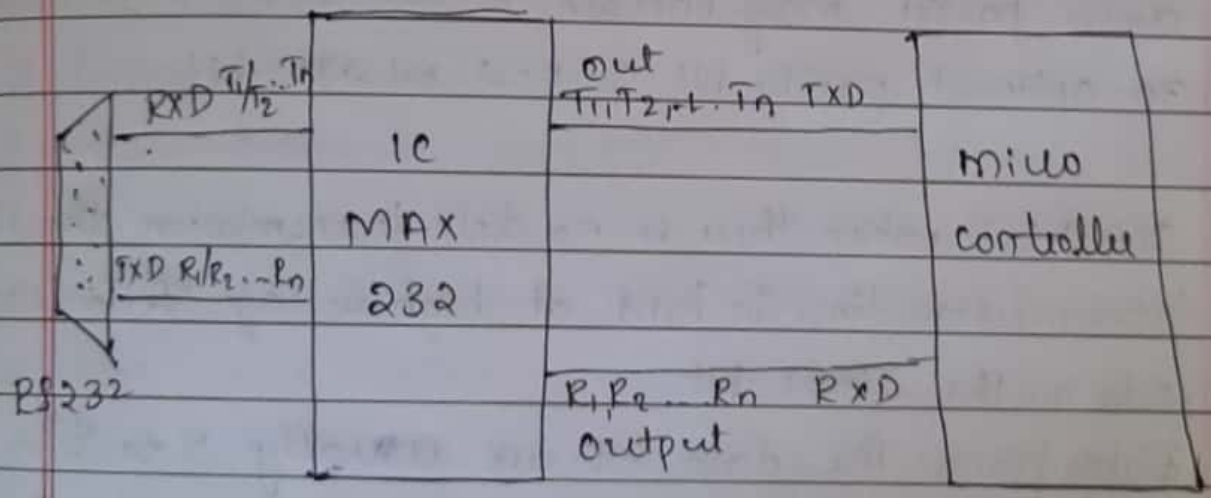
Before anything else happens the modem must evaluate the quality of the time negotiate error control protocols & data compression that they can both recognise & work out what the most suitable connection speed should be based on the conditions. This process is called a Handshake.

29 Explain RS232 connection with a MC

→ Serial devices collect data from sensors & need to send it to another unit like a computer for

further processing. Data transfer/communication is generally done in two ways. is fast & use more number of lines.

Serial communication on other hand use only one or two data lines. transfer data and is generally used for long distance communication In Serial communication In Serial communication the data is sent as one bit at a time



An important parameter considered while interfacing Signal port is the Baud rate which is the speed at which data is transmitted serially. We can be set to transfer and receive signal data at different baudrates using software instructions.

RELIABLE
Page No.:
Date: / /

20. Explain the frame format in UART communication.

START	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	P_B	STOP

Baud rate: It is a data transmission refers to the number of symbols transferred per second. A symbol is a group of a fixed number of bits

Data framing: UART transmit data in packets Each data packet may contain start Bit, 5-9 data bits an optional parity bit & 1-2 bit STOP bit

START bit: when there is no data transmission the UART transmission line is held at high voltage. To transition acts as the START bit

Data frame: The data bits are usually 5 to 8 in number. If no parity bit is used, it can be 9 bit long. In general case the LSB to the data is transmitted first

Parity Bit:

The parity bit is used to indicate the change in data transmission. The reason for the change in data mismatched baud rates, electromagnetism or long distance data transfer

STOP bit: To mark the end of data packet the sending UART drives the data transmission line from low

voltage to a high voltage for a minimum of two bit duration.

8] Explain the differences.

a) Serial v/s parallel

Serial	parallel
* data is transmitted bit wise in a single line	* data is transmitted simultaneously through group of lines
* Data congestion takes place	* No data congestion
* Low speed transmission	* High speed transmission.
* Implementation of serial lines is not an easy task.	* Parallel data lines are easily Implemented as hardware.
* no crosstalk problem	* crosstalk creates interference between parallel lines.
* The bandwidth of serial wire is much higher.	* Bandwidth of parallel wire is much lower.

b) Analog v/s digital

Analog	digital
* Transmitted modulated signal is analog in nature.	* Transmitted signal is digital ie train of digital pulse
* amplitude frequency or phase variation in the transmitted signal represent the info of message	* Amplitude width or position of transmitted pulse is transmitted in the form of code words.

RELIABLE
Page No.:
Date : / /

* Noise immunity is poor for FM & PM	* Noise immunity is excellent
* coding is not possible	* coading techniques can be used to detect & correct errors
* FDM is used for multiplier	* TDM is used for multiplexing
* Analog modulation systems are AM, FM, PM, PAM, PWM	* Digital modulation system are PCM, PM, ADM, DPCM.

c) Synchronous v/s Asynchronous.

Synchronous	Asynchronous.
* Communicated in real time	* not communicated in real time
* Creates interrupt in a cookie	* Eliminates interrupts
* Sends & Receives data on same clock frequency	* Sends and Receives data on diffent clock frequency
* faster	* Slower.
* There is no overhead of extra Start & Stop bit	* uses Start & Stop bit
* uses constant time interval	* Uses random or irregular time interval
* Used in chatrooms & Video	* Used in Emails.

Page No.: RELIABLE
Date : / /

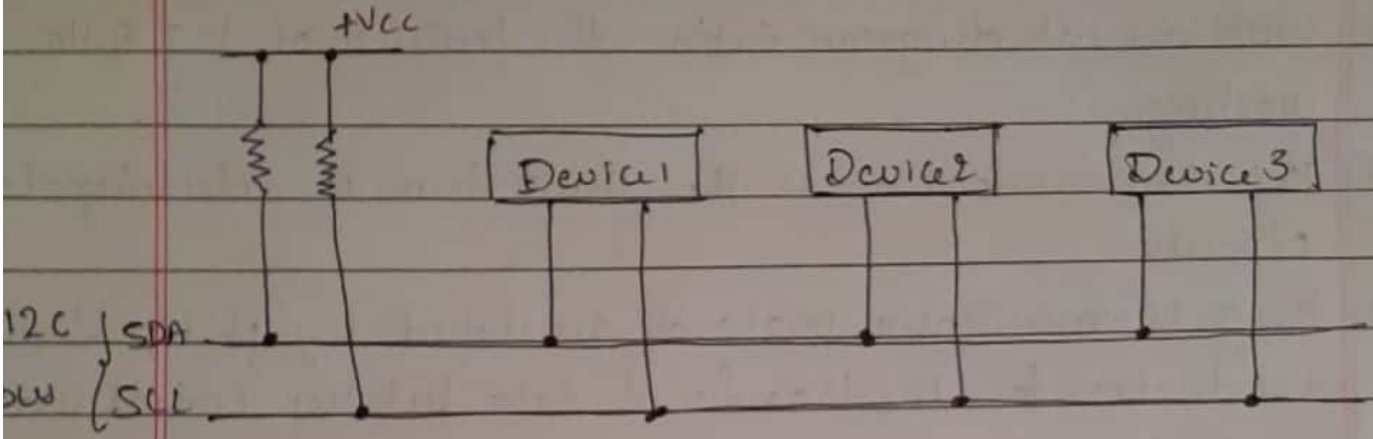
24 With a neat diagram Explain the features of I2C & its working.

* I2C communication is the short form of Inter Integrated circuits.
* Its a communication protocol developed by philips Semi conductor for the transfer of data between central processor and multiple ICs on the same circuit board using just two common wires.

Features:

> only 2 common bus lines (wires) are required to control any device/IC on I2C network
> uses 7 bit addressing system to target specific device/IC on I2C bus.
> I2C network are ~~usually~~ easy to scale. New devices can simply be connected to the two common I2C bus lines

• I2C bus consists of just two wires and is named as Serial clock Line (SCL) & Serial data line (SDA). The data to be transferred is sent through SDA line & is synchronised with clock signal from SCL. All the devices on the I2C network are connected to the same SCL & SDA lines



- Both the I2C Bus lines (SDA, SCL) are operated as open drain drive. It means that any device on I2C network can drive SDA and SCL low, but they cannot drive them high. So a pull up resistor is used for each bus line to keep them high by default.

Master and Slave devices :

The devices connected to I2C bus are categorized as either master or slave. At any instant of time only a single master stays active on the I2C bus. It controls the SCL clock line and decides what operations is to be done on the SDA data line.

All devices that responds to instructions from this master device are slaves. For differentiating between multiple slave devices connected to the same I2C bus, each slave device is physically assigned a permanent 7 bit addres

Working

An I2C communication is initialised by a master device either to send data to a slave device or to receive data from it.

* Sending data to a slave device.
  - The master device sends the start condition.
  - The master sends the 7 address bits which corresponds to the slave device to be targeted
  - The master device sets the read/write bit to '0' which signifies a write.
  - Now 2 scenarios is possible:
    - If no slave device matches with the address sent by the master device, the next Ack/NAck bit stays at '1' (default). This signals the master device that slave device identification is unsuccessful.
    - If slave device exists with same address as the one specified by master device, that ~~next ACK/NACK bit stays~~ at '1' (default) the slave device sets ACK/NACK bit to '0' which signals the master device that slave device is successfully targetted.

* Reading data from slave:
  - The master device sets the Read/write bit to '1' instead of 0. which signal the targeted slave device that the

master device is expecting data from it.

- The 8 bits corresponding to the data block are sent by the slave device and the ACK / NACK bit is set by the master device.
- Once the required data is ~~retrived~~ received by the master device, it sends a NACK bit. Then the slave device stops sending data & releases the SDA line