ON SCRATCH PAPER WRITE BINARY 0 to F!

- An **embedded system** is an electronic system controlled by one or

more computers/microprocessors that are internal to the system. That

is, the system is not thought of as a computer, but something else.

- An example of a system that needs to be a “realtime” system and what

the functionality of it is in real time: **Air traffic control system, the**

**current location.**

- A **microcontroller** is a “computer on a chip”

- Who makes the processor discussed in this class? There are two

answers: **Microchip Founded 1987.**

- A **real-time** system must insure that a task is completed within a

certain time span.

- Name five things discussed in class, which must be considered when

designing an embedded System. **Real-Time Execution; Physical Size**

**and Environment; Power Consumption; User Interface; Multirate**

**Operation; Cost; Memory Needs; Hardware versus Software**

**-**Old chip was: **Mips32 M4K Core**

-In a time span: **real-time system**

-Diagnostic PIC32: **JTAG**

-Functions: **CTMU, Timer, PWM, ADC, UART, SPI, RTCC, Compare**

-CTMU External Input Pins : **11**

-UARTs: **2**

- The HC12 has two 8bit accumulators, **A** and **B**, which can be ganged

together

- The HC12 has two **8**-bit accumulators, which can be ganged together.

- The HC12 has at most **128**–kB of flash on board.

- **\***What is the size of the registers in (in bits) of the HC12 processor?

(What size are the standard addresses (in bits) of the HC12 processor?)

i.e. “The HC12 is a \_\_\_ -bit machine.” **16**

- The HC12 has a maximum **8** MHz bus clock.

- The HC12 has a **16**-bit stack pointer.

- The program counter in the HC12 is **16** bits.

- The HC9S12 was an improvement on the HC12 and has a maximum

**25**-MHz Clock.

- The HC9S12 has at most **512**-kB of flash on board.

- The HC12 has two [a]-bit registers, X and Y. **16**

- How is SRAM generally used in a system? **Lvl1 cache, Lvl 2 cache, NVRAM**

- What type of Memory is used to store variables in a(n) (embedded) system? **RAM, Volatile, SRAM, DRAM**

- What is the advantage of using DRAM over SRAM? **Lower power at use, Higher Density, Less expensive.**

- **\***What type of memory is made out of transistors? (What type of RAM is made out of transistors?) **SRAM**

- Which of the following is volatile? **SRAM, DRAM, SDRAM**

- What type of memory is used to store the Stack in an embedded system? **RAM, Volatile, SRAM, DRAM**

- Which type of memory is generally used to store system states in an embedded system? **ROM, NonVolatile, PROM, EPROM, EEPROM**

- What type of memory is generally used to store Program Instructions in an embedded system? **ROM, NonVolatile, PROM, EPROM, EEPROM**

- What type of memory is generally used to store Tables or Images in an embedded system? **ROM, NonVolatile, PROM, EPROM, EEPROM**

- What is the advantage of using SRAM over DRAM? **Lower power consumption (at rest), Simpler to implement, Faster, Can be placed on same die as processor logic**

- What type of memory is generally used to store User Settings in an embedded system? **ROM or NonVolatile or PROM or EPROM or EEPROM**

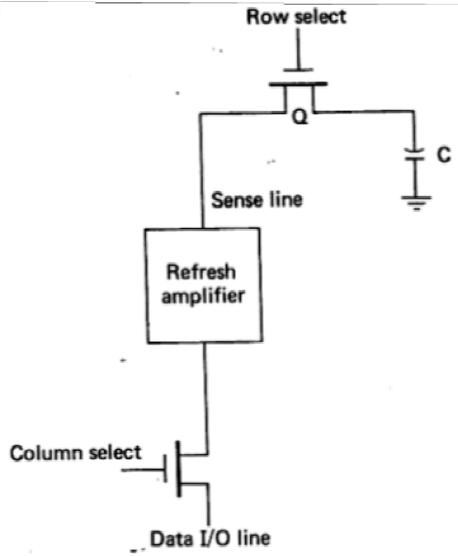
- Which of the following are nonvolatile?

**EEPROM, EPROM, NVRAM, OTP ROM,**

**Flash, Masked ROM, PROM.**

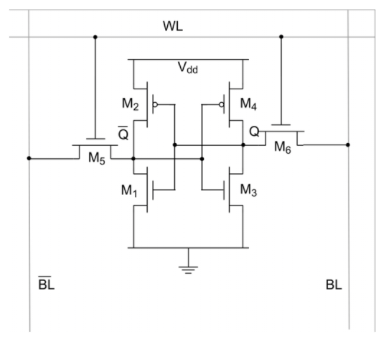
- Which type of RAM stores its data using small capacitors? **DRAM**

- Which type of RAM is the following memory cell found in?

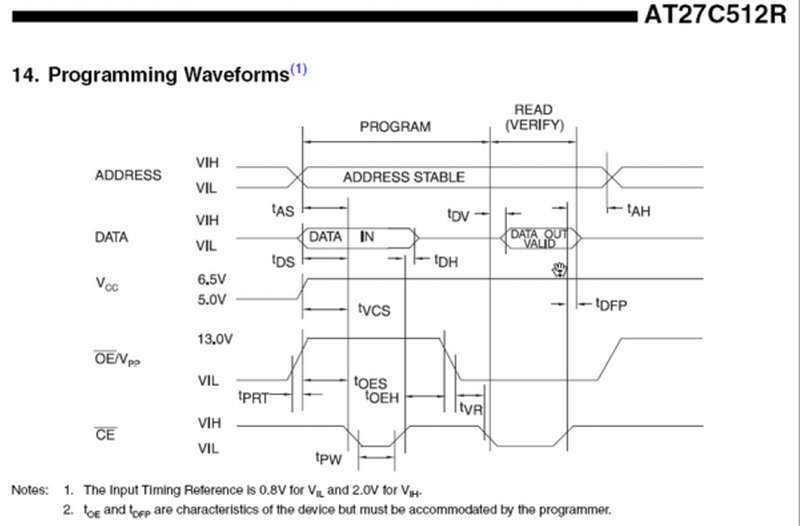
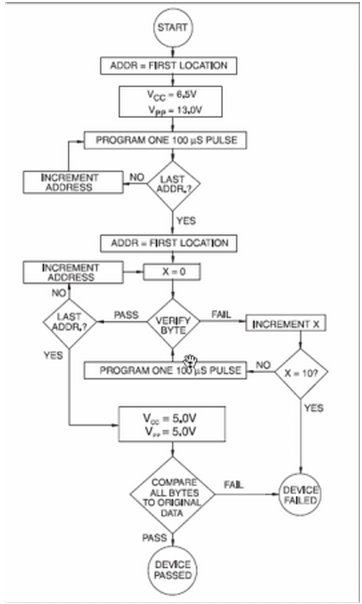


**DRAM**

- The memory cell shown below is used in what type of memory?



**SRAM**



- Flowchart, how many times does the programmer try to program a difficult cell before it quits? **10**

- Timing diagram, how long must programmer wait to insure the DATA is valid after the EPROM is told to output the cell data written to it? **tDV**

- Flow chart, how does the programmer program the EPROM? **Programs all the cells, then checks each one for validity**

- Timing diagram, at what point in time does the EPROM programmer read the DATA to be verified from the bus? **The second rising edge of CE#**

- Timing diagram, how long must the DATA be stable before the EPROM latches the data into memory? **tDS**

- Timing diagram, how long must the ADDRESS be stable before the EPROM latches data into memory? **tAS**

- Timing diagram, what signal tells the EPROM to output the cell data which was written to it? **CE#**

- Timing diagram, at what point in time does the EPROM read the DATA it is to program into its cell? **The first rising edge of CE#**

- Timing diagram, how long must the programmer leave the ADDRESS on the bus after they have read the data to be verified? **tAH**

- Timing diagram, what signal determines when the EPROM reads the data to be programmed into the cell? **CE#**

New Section Name four different things that you generally find on a microcontroller project board. **Reset Switch; Com Ports; LCD Display; 7Segment Display; LEDs; Keypad/Switches; Amplifiers/Op Amps; OptoIsolators; Relays; SCRs/Triacs; Buzzers; Potentiometers; DACs; Dip Swiches**

- What’s the name of an electromechanical device that is used to switch higher voltage and/or higher current devices? **Relay/SCR/Triac**

- What field does the ‘0038’ reside in the s-record shown below?Macintosh HD:Users:brandonstem:Dropbox:Screenshots:Screenshot 2015-10-05 08.59.02.png

**S=Start Code, 1=Record Type, 11=Byte Count, 0038=Address, [42]=Data, 42=Checksum**

- What is circuitry that is used to interface two different chips together called? **Glue logic**

- What kind of IC is used to isolate a low voltage microprocessor circuit from a higher voltage motor circuit? **Opto-isolator**

- What is the name for a program which takes C code written on a computer and creates code for a microprocontroller? That is, it creates for a processor other than the one it runs on. **Cross Compiler**

- What is the name for software and/or hardware which allows you to start and stop execution and examine the state of the processor while it is executing your code? **Debugger**

- What is the name for the file generated by a compiler to run on a Motorola/Freescale processor? **S19/S-record**

- What might a potentiometer be used for in an embedded system? **Used to as a volume control in audio devices. Act as a variable resistor**

- What is a relay used for in a circuit? **Relays can control current flow without having hardware/physical device that can degrade of time**

- What are dip switches used for in an embedded system? **For setting up various configuration options**

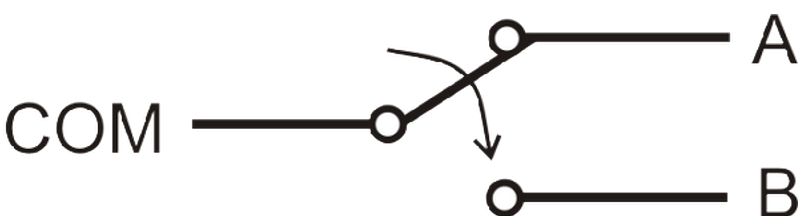
- How is a “momentary switch” different than a regular “on/off” switch? **Momentary is like a push button, on/off switch is like light switch**

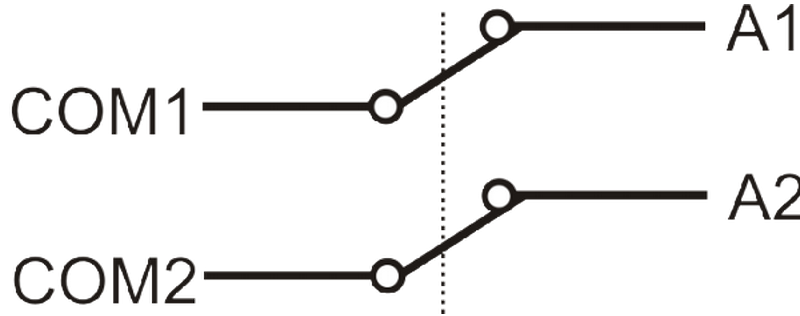
- What do you call a switch that is normally off, and must be held down (depressed) to be on? That is, once it is released it is off again. **Push Button Switch/ Momentary Switch**

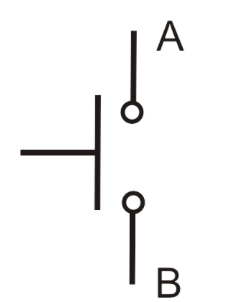
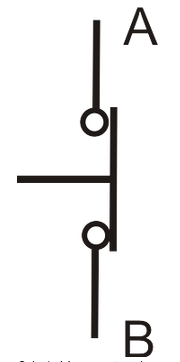
- What does SPST on a relay stand for? **Single Pole Single Throw**

- What does DPDT on a relay stand for?  **Double Pole Double Throw**

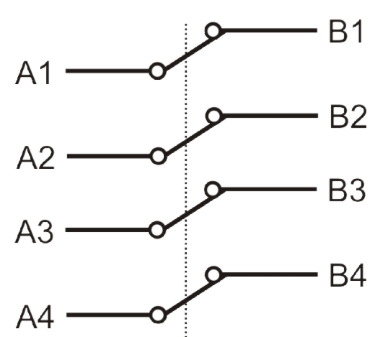
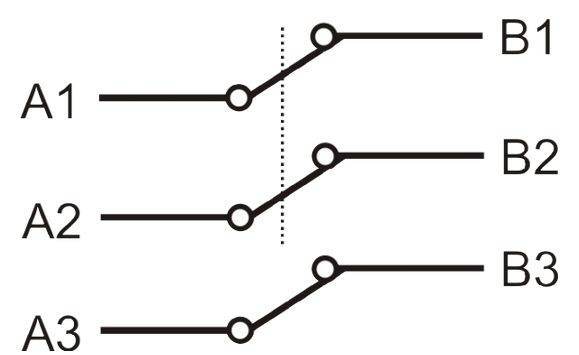
- What does SPDT on a relay stand for? **Single Pole Double Throw**

 **Single Pole Double Throw**

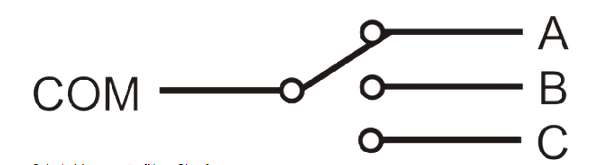
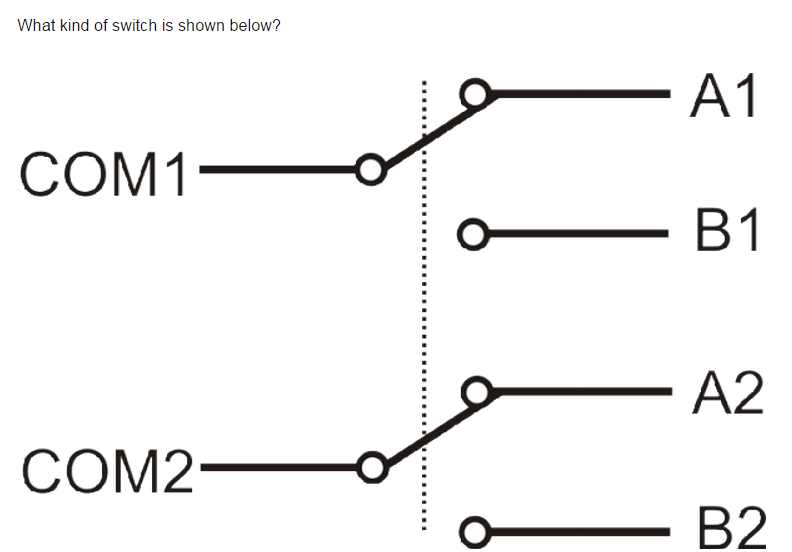
**Double Pole Single Throw**



**Normally closed momentary switch Normally open mom. switch**



**Triple Pole Single Throw Quad Pole Single Throw**



**Double Pole Double Throw Single Pole Triple Throw**

- The “256” in the part number MC9S12DP256B refers to the amount of **memory** contained in the chip.

- The term **glue logic** refers to the added circuitry necessary to interface two or more chips together.

- The **flash** memory in the chip is used to hold code instructions for the chip.

- What signal from the MC9S112 is used in conjunction with the R/W# line to determine whether or not 8 or 16 –bit data is being read? **LSTRB**

- Which of the following is/are not found on the MC9S12? **Piezo Electric Buzzer, Digital to Analog Converter, Temperature Sensor**

- What type of data/information is stored at the upper end of the MC9S12 memory map? **Interrupt Vector Table**

- The [a] memory in the chip is used to hold code instructions for the chip. **Flash**

- What is a compiler called which generates code for a process other than the one it is running on? **Cross Compiler**

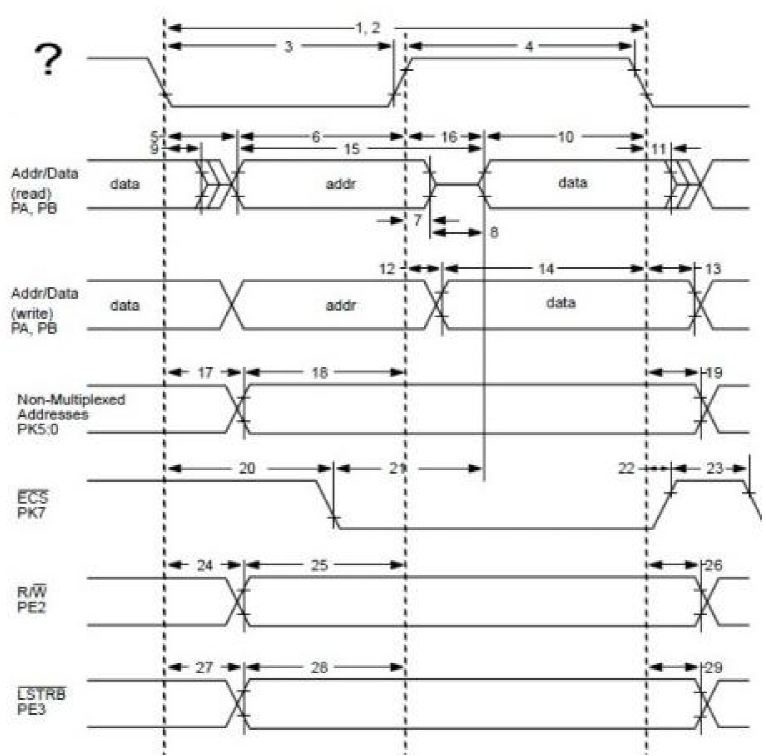
- What signal from the MC9S12 is used to determine whether or not a memory chip is being read from or written to? **R/W#**

- What type of memory is located at the very beginning (lowest addresses) of the MC9S12 Memory Map by default? **Registers**

**-** What type of memory in the 9S12 would you use to store program setting in? **EEPROM**

- What signal from the MC9S12 is used for memory reads when external memory is used, but is not used in single chip mode? **LSTRB**

-What signal is shown at the top of the timing diagram? (As indicated by a ‘?’)

** ECLK**

Clear PTH: PTH = PTH & 0x52;

Analog: ANSELxSET = 0xA000;

Digital: ANSELxCLR = 0xA000;

Change Notice Interrupt

Disable: CNENxCLR = hex;

Enable: CNENxSET = hex;

Internal Pull Down

Enable: CNPDxSET = hex;

Disable: CNPDxCLR = hex;

Internal Pull Up

Enable: CNPUxSET = hex;

Disbale: CNPUxCLR = hex;

TRISx: 1-> in 0-> out

PORTx = read (value of pins)

n = PORTx & 0xA234;

LATx = write

Zeros: LATxCLR = hex;

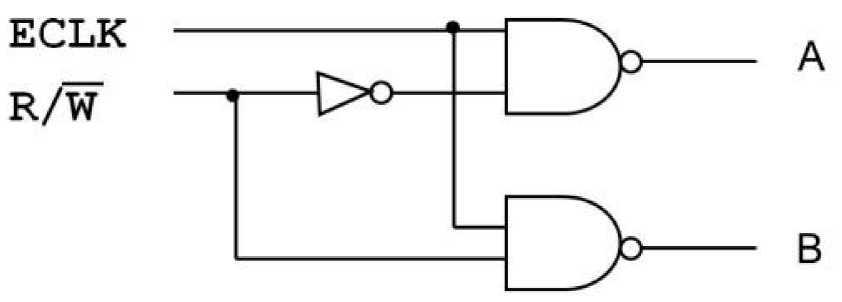
Ones: LATxSET = hex;

Outputs Open Drain:  
ODCxSET = hex;

Outputs Pull Push:  
ODCxCLR = hex;

Port[0] is right most port

- In the glue logic below the signal A is [a] and the signal B is [b].

** A =** **WE#; B = OE#**

**Battery\*(AmpHour/24\*30\*months or 24\*365\*year) = Avg Draw**

**Need(x) + Don’t Need(1-x) = Avg Draw**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

- What is the purpose/function of a pull-down resistor? **Pulls an input down to a zero by latching it to ground**

- Give an advantage and disadvantage of using isolated(port-based I/O. **Pro – Simultaneous memory and I/O. Cons- Additional pins needed for address, data, controls; additional instructions needed to access the I/O bus**

- What latch chip was used in class to **write** to I/O devices? **74374**

- **\***(What does the following line of C instruct the compiler to do? What does it allow the programmer to do? (What does unsigned char\* do?) (What does volatile mean?)

#define PTH \*(volatile unsigned char \*)0x260

**Variable can be changed from outside the program (external device can drive high/low)**

- Give an advantage and disadvantage of using memory-mapped I/O. **Pro – many operations, no addition pins or instructions necessary; Con- I/O error on a memory map cannot be caught and results in program crashing, less memory can be addressed.**

- Give an advantage and disadvantage of using “linear select addressing of I/O. **Pro – Simple selection logic and one port per address-line; Con – Wasteful Addressing**

- Why is a latch needed to store the address from the 9S12 when communicating with an I/O device? **Bus is multiplexed so address must stored while data is put on the bus**

- Why is the 373 Latch’s G input connected to 5 volts? **The latche’s G input is not-ed so it has to connect to 5 volts to make it a zero.**

-What is the difference between the 74-373 and the 74-374? **74373 uses an enable, 74374 latches values on the rising edge of a clock**

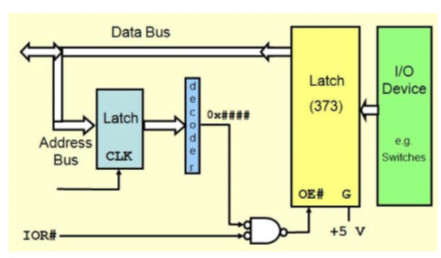
- What latch chip was used in class to **read** from I/O devices? **74373**

-\*What is a typical size/value of a pull-up/ pull-down resistor? **4.7KOhm - 10KOhm**

-The ISA does not support serial I/O directly. Why? **Because everything starts out as parallel within the CPU**

- Everything done in the CPU is done in parallel (not serial). Why? **This is done because there are plenty of jobs that computer has to do and most all of them are performed individually. Therefore, parallelism makes everything WAY faster and other programs do not have to wait for another to complete before it executes.**

- Why is it necessary to have the Latch on the left (light blue)?



**It uses the clock pulse to latch the values from the address bus to the decoder. Prevents all values from being latched at once.**

- An open drain output allows for “wired-Oring” or pins. **True**

- This register can be used to detect overloads or shorts at port pins. **PTln**

-This register can be used to internally connect a resistor to form a port pin to either Vcc or Vdd if the corresponding pin is set to input. **PERn**

- In order to know which pin an interrupt has occurred on, this register must be read. **PIFn**

- A [a] resistor is generally used to make an input pin’s default value 1. **Pull up**

- This register can determine whether the pull resistor is “pull-up” or “pull-down”. **PPSn**

- This register can determine whether the interrupt is detected on either the rising edge or falling edge of a signal. **PPSn**

- Which register is written to in order to change the values of a certain port’s pins? **PTn**

- What Port only has 4 pins instead of 8? **J**

- Which register needs to be configured before ever outputting a value to a certain port pin? **DDRn**

- This register determines whether or not a transition on a port pin will produce an interrupt. **PIEn**

- The ISA level directly supports serial instructions. **False**

- Which register determines if port pins will be used as input or output? **DDRn**

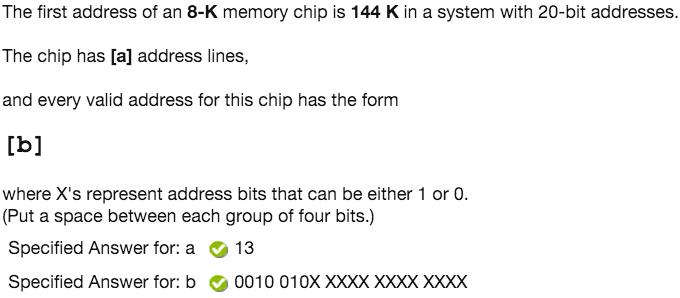
- Which register determines if port pins will act as an OR gate when its pins are tied together, or not? **WOMn**

- This register reads the state of the register buffer if reading a pin is set to output. **PTn**

- This register tells whether or not an interrupt has occurred a specific pin of a port. **PIFn**

- This register can be used to limit the amount of current supplied to a circuit connected to the corresponding port. **RDRn**

- This Register ignores data written to it if its pins are set up for input. **PTn**

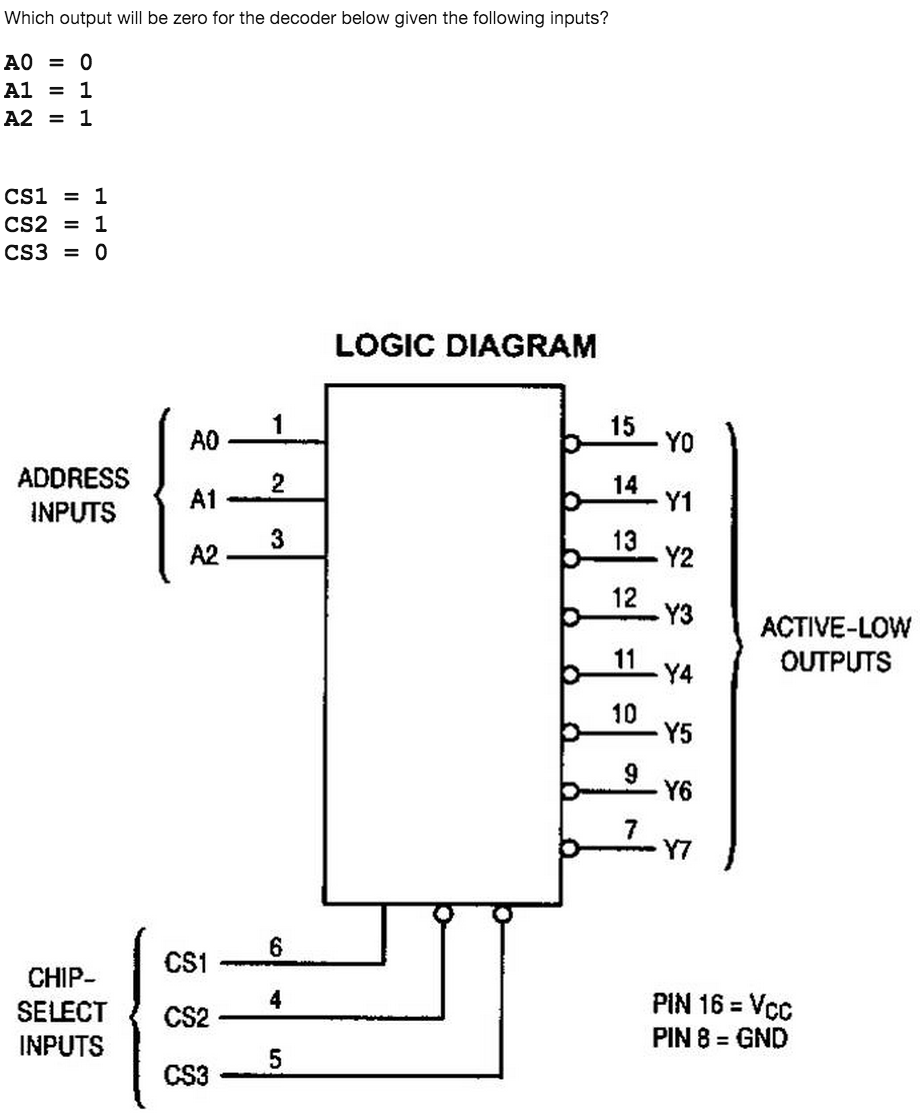


- 2^? = size of memory chip => Address Lines (a)

- Everything beyond address line is an X

- Using values below, add up to get address number, place 1 where number is used (All values are in K(thousands))

- 19=512,18=256,17=128,16=64,15=32,14=16,13=8,12=4,11=2,10=1



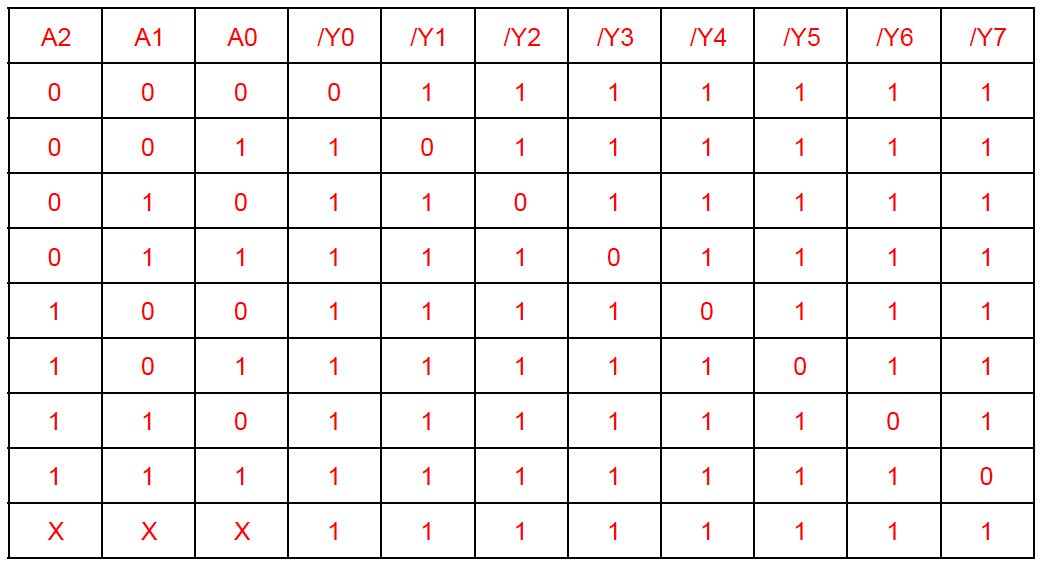
-Match A’s with table, CS1=Y0, CS2=Y1, CS3=Y2

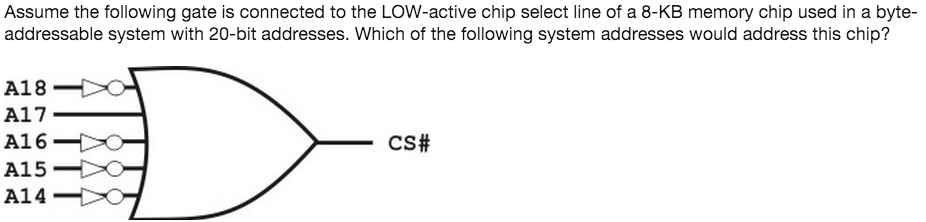
-If they all match, find the 0 in the line

-If they don’t match, no outputs

-Usually….

NO OUTPUT





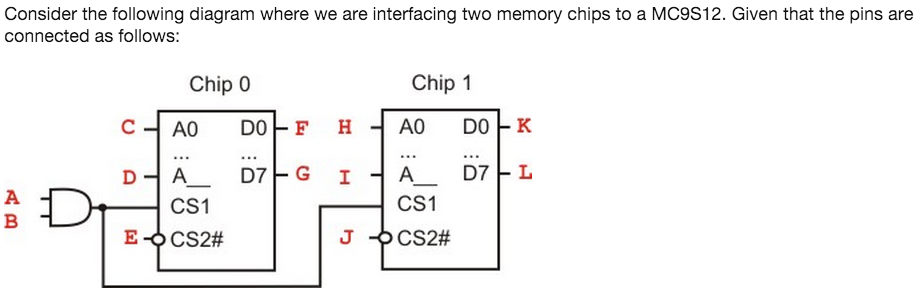
Watch for Skips

- Determine low active or high active, pay attention to not’s

- Fill in appropriate 1’s or 0’s to get correct output on each line

- If line is excluded, put an x => x can be 1 or 0

- Convert A19-A0 to hex



a. 2^(D-10)

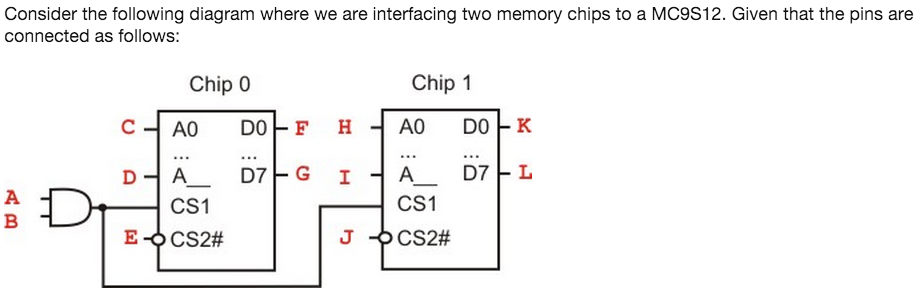
b. FG and KL=>D0 to D7 is low order chip (F = D0, G = D7 -> low)

(F = D8, G = D15 -> high)

c. A19-A0=> Put 1 where A and B are, 0’s otherwise, convert to HEX

d. A19-A0=>Put 1 at A and B, at D put 1’s onward, otherwise 0’s

e. Big endian if LSTRB goes into low-order chip (high –> little endian)



High/Low is Given

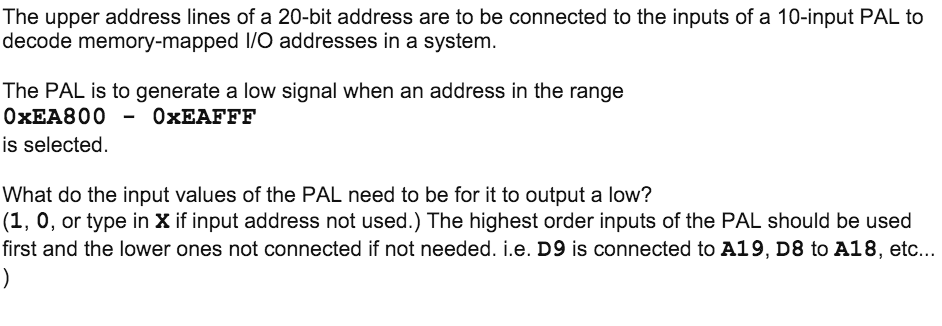
- C and H are A1

- D and I=> take ending address, covert to binary, where bit’s are all 1’s, that starting position is D and I

- E and J=>If big endian **LSTRB** goes into Low order chip, Little endian LSTRB goes into high order chip, **A0** goes into opposite chip

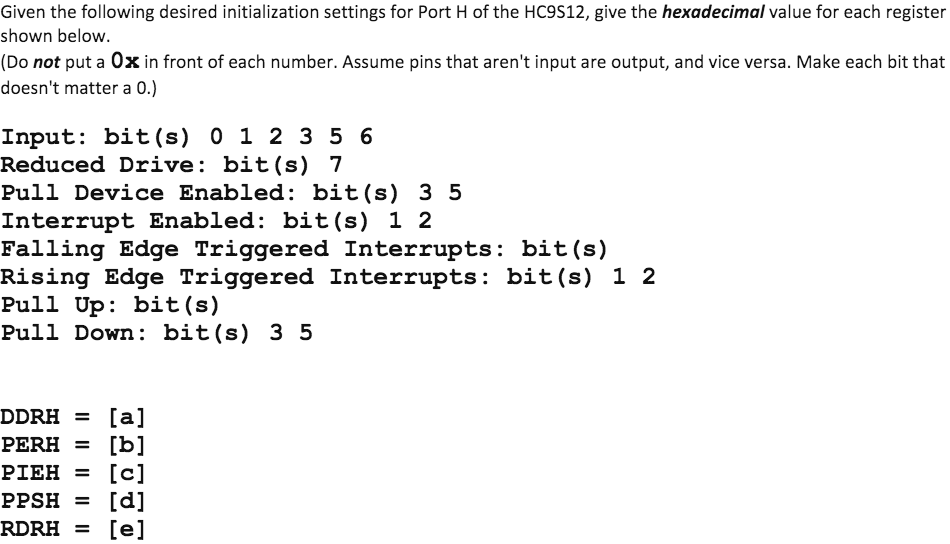
- FG and KL=>Low order chip is D0-D7, other is D8-D15

-Chip 0 = 2^(D-10)



- Convert both to binary=>Write starting address on top of ending

- Put x where they start to differ and onward



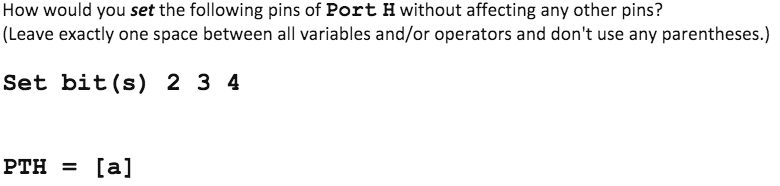
- DDRH=Either output or inverted input bits

- PERH=Pull Device Enabled Bits

- PIEH=Interrupt Enabled Bits

- PPSH=Rising Edge Triggered Interrupt bits and pull down bits

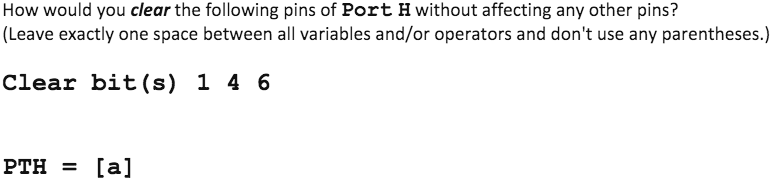
-RDRH=Reduced Drive Bits



- Convert to Binary

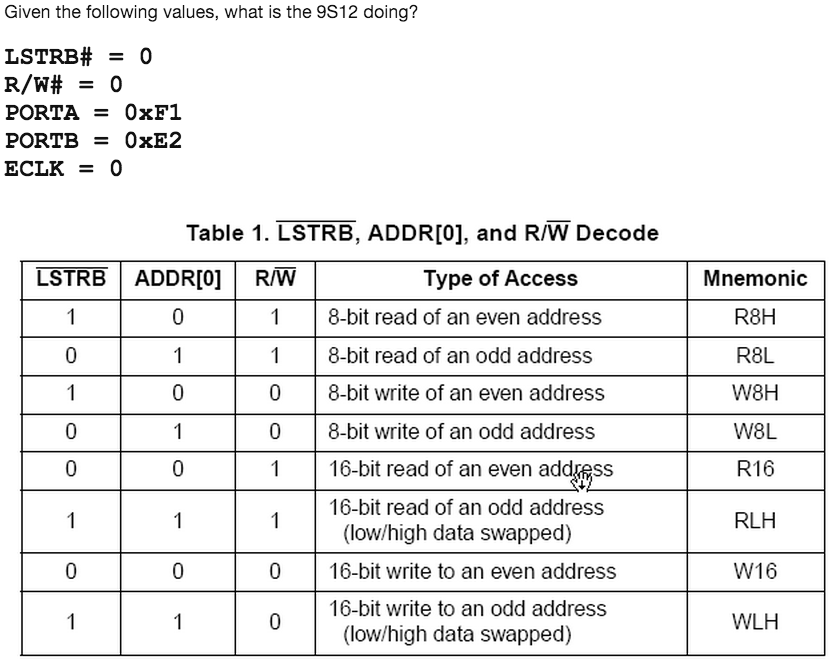
- If set, OR with PTH PTH | 0xSomething

- If clear, AND with PTH PTH & 0xSomething

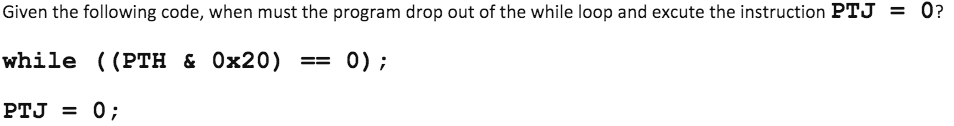


- Convert to Binary=>Invert the binary number=>Convert to hex

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 Convert Port B to Binary => the rightmost bit is value of ADDR

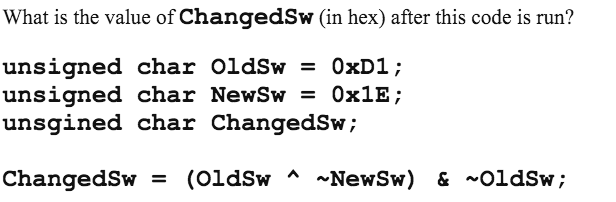
- Match up other columns => ECLK=0, preparing ECLK=1,performing



- Convert to binary

- If while loop = 0=> Every bit that is a 1, put Port[x]=1

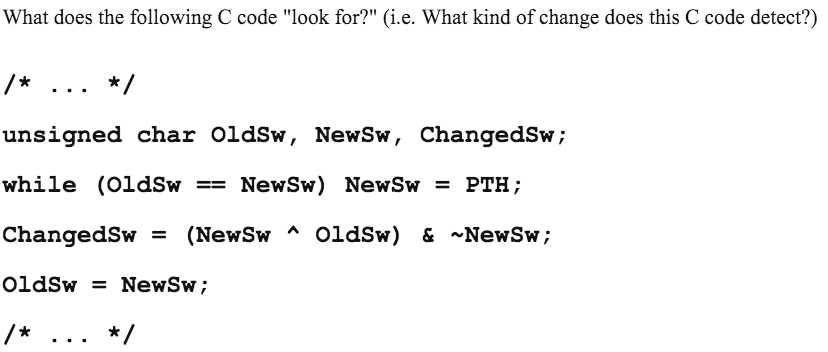
- If while loop = 1=> No possible solutions



- Convert all to binary

- When raising binary to binary => bits are same, 0; bits are different, 1

- Follow equation



Look where turned to 1

- Use 1010 for Old->Use 1001 for new->Follow code, see what changed

**KEYPAD**

- Rewrite Keypad as

- Circle keys pressed->0’s if pressed->1’s otherwise (in form of keypad)

- For bitshifting->convert across to binary (Nott if needed)

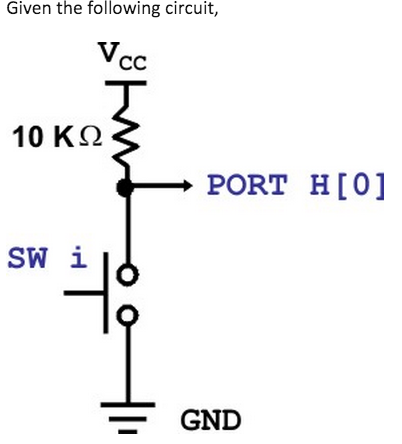
- Otherwise look at keypad and determine where button was pressed

- Single button in row, it’s that one; Two single buttons in two different

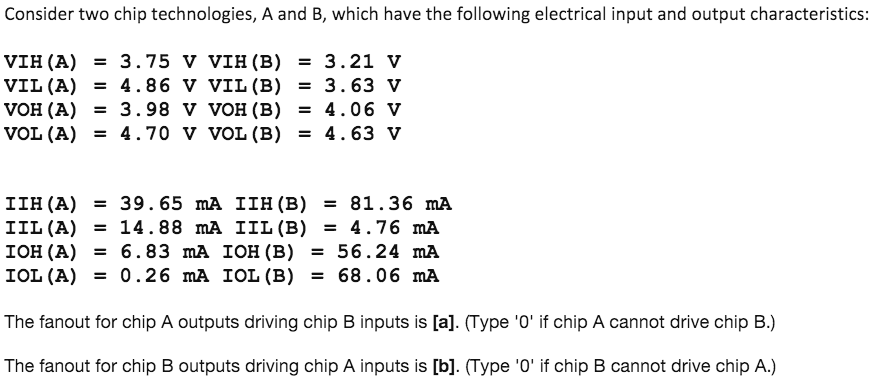
rows, it’s the top button; Two buttons in same row or none pushed, it’s 0

Switch on bottom

Switch on Top



**---------------------------------------------------------------------------------------**



Round Down

Only one will be 0

- First look at II’s->A to B make sure Max A>Min B if not A = 0

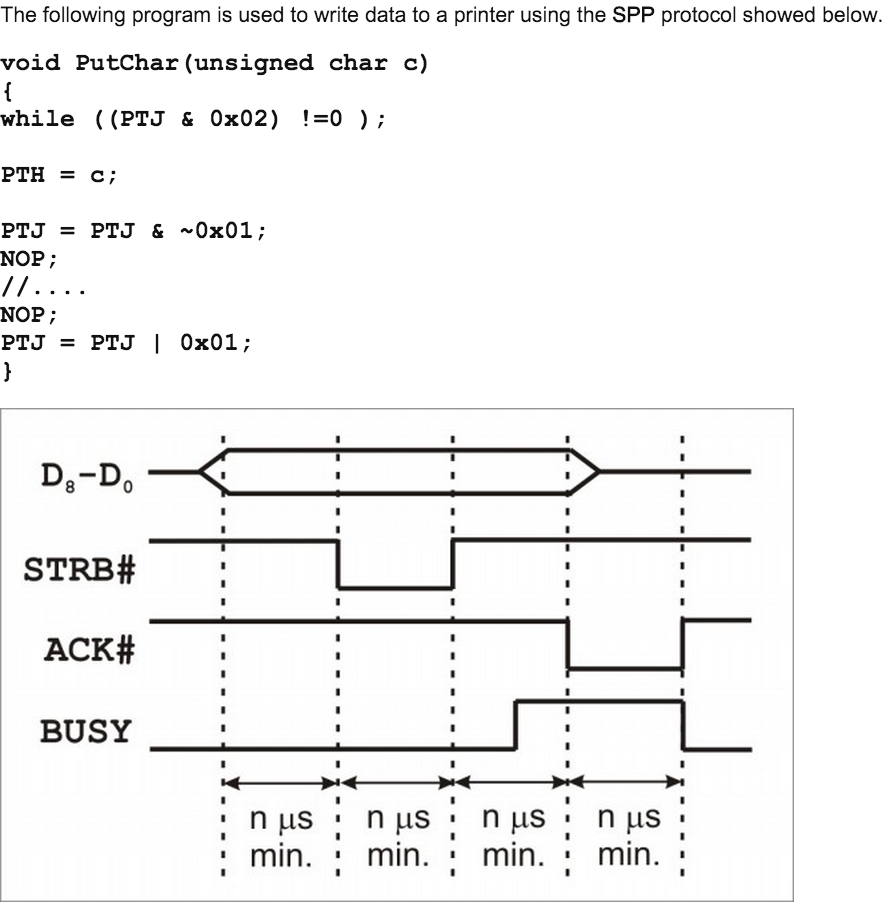
- Repeat for B to A-> Max B>Min A if not B = 0

- Repeat steps for IO’s

-If pass both test; A to B get minimum of IOH(A)/IIH(B) and IOL(A)/IIL(B)

- Whichever is min, that is = A -> Switch A and B’s for B

**----------------------------------------------------------------------------------------------**



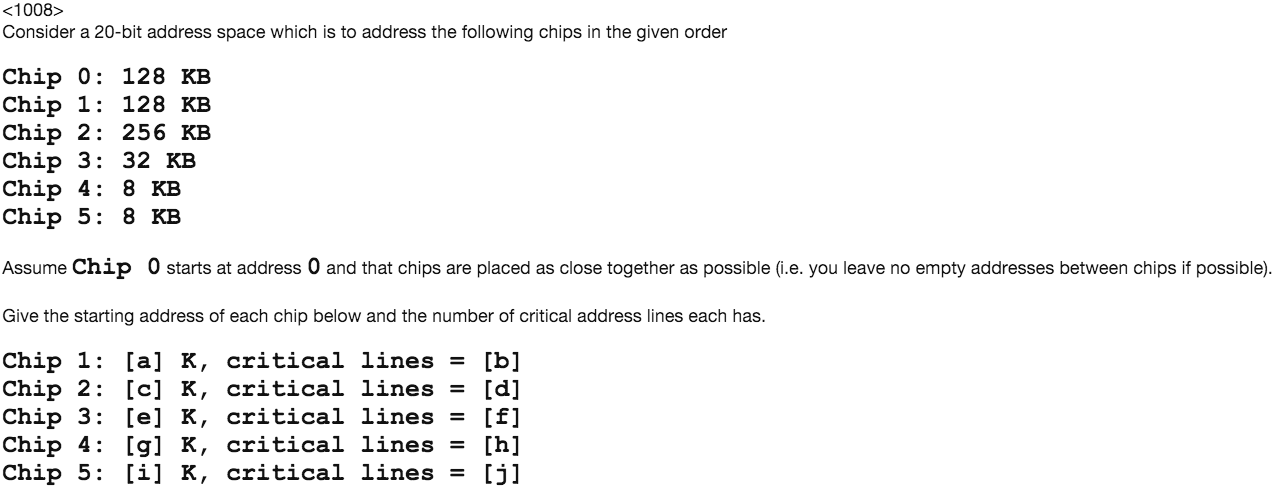
- See how man blocks STRB# = # (in this example 1)

- (( #\*N\*(# of Mhz))/(clock ticks))-1 (don’t use x10^-3 or any multiplier)

-round up

**----------------------------------------------------------------------------------------------**

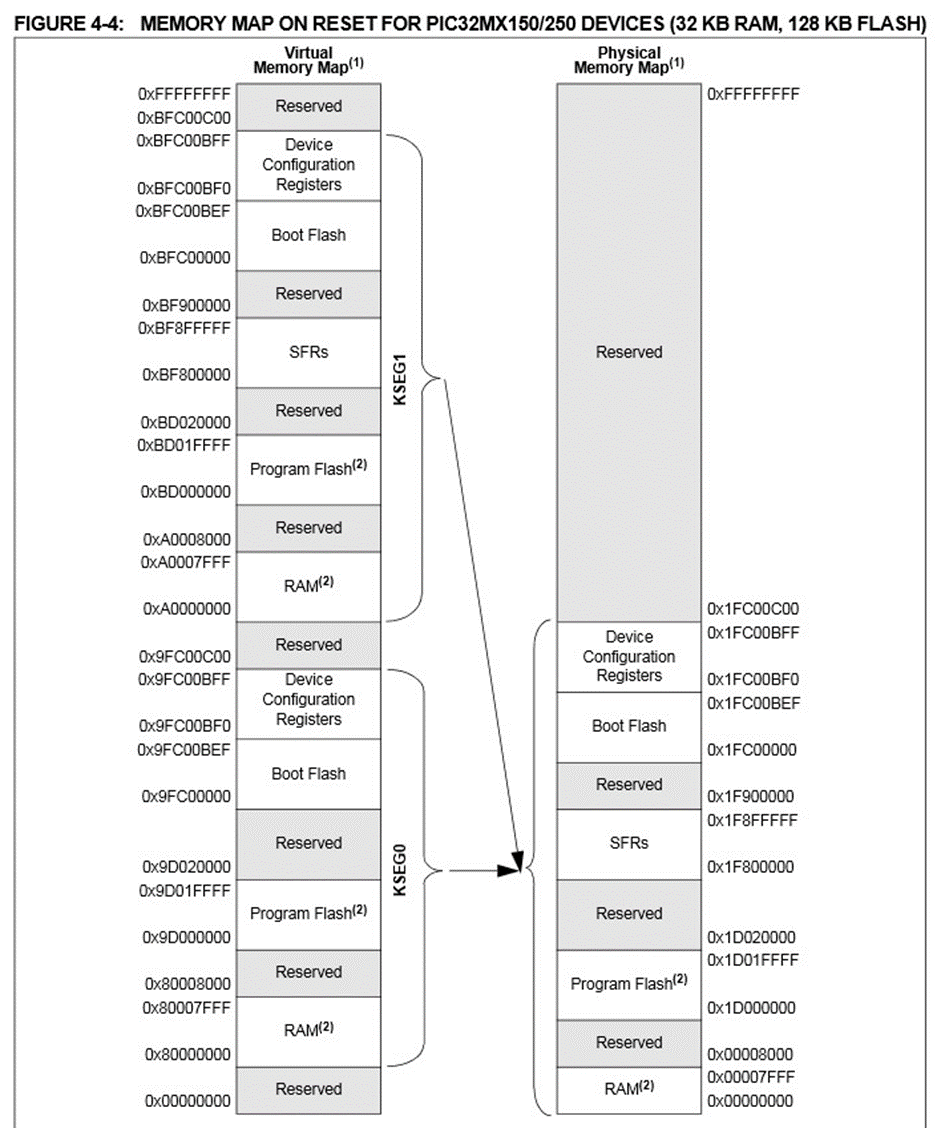
**ADDRESS DECODING**



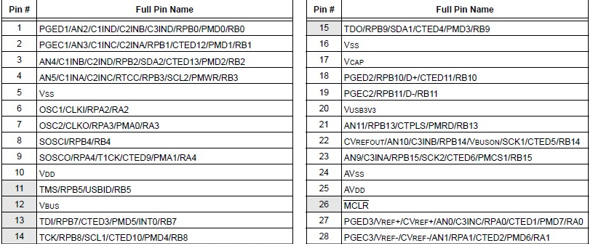
Remember: address/size = int

Check 0s, 1s always \*

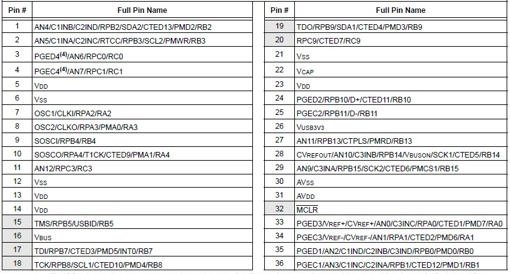
Make a table



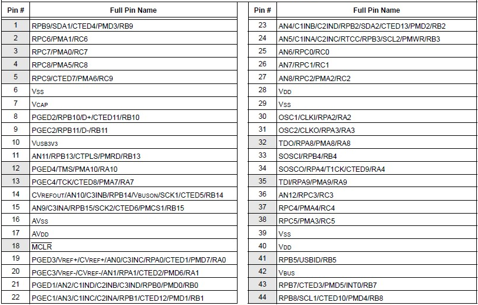
28 OFN

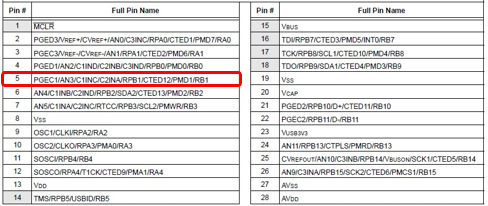


36 VTLA



44





28 QFN

