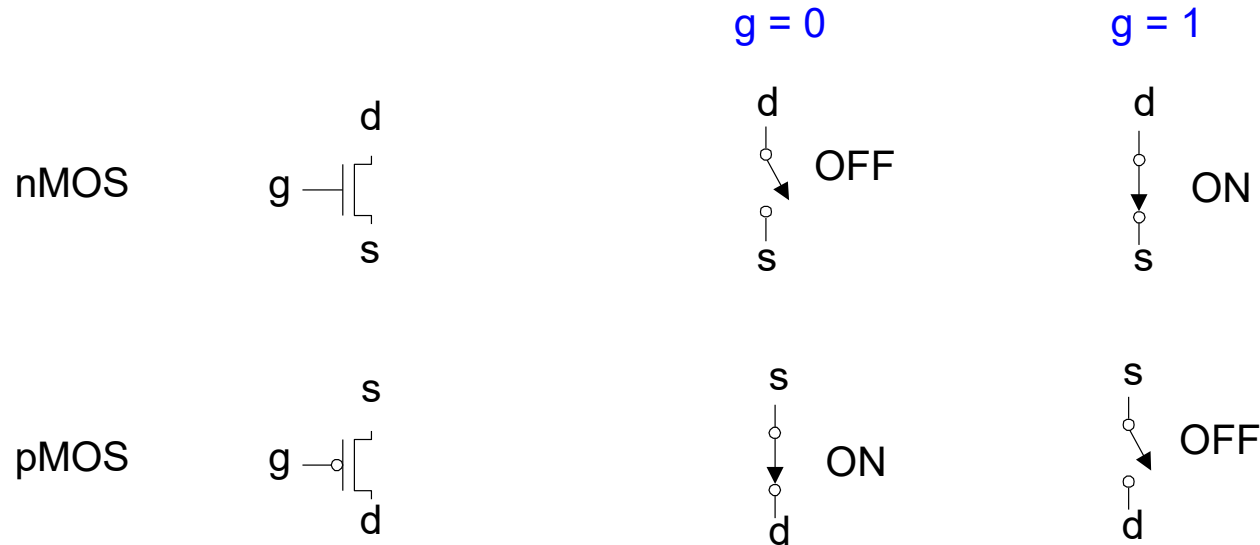


NCS 362: Embedded Systems

- ~~Intro to Embedded System.~~
- ~~Review of Electronics Part (KCL, KVL, Parallel/Series Resistance, DAC and ADC).~~
- Review for Digital Systems (Binary Number, Logic, MOS Implantation, Computer Architecture).
- Intro to Programming Concepts (Structure and Concurrent).

Binary Information Implemented with MOS transistors

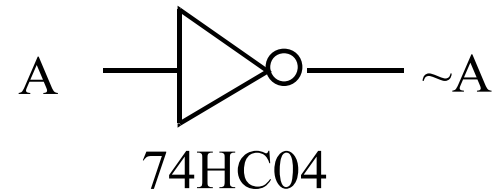
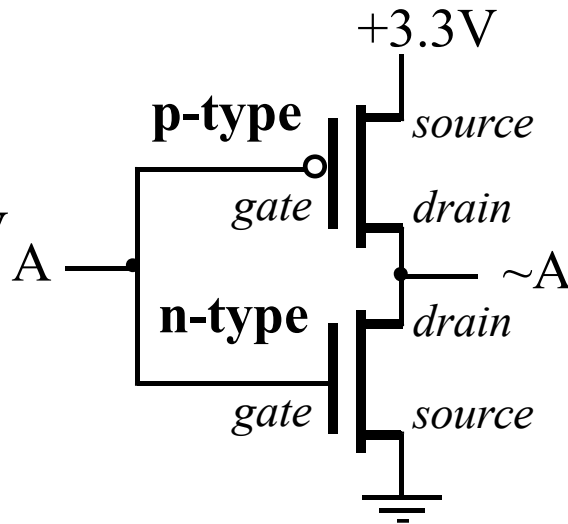


nMOS: pass good 0's, so connect source to GND

pMOS: pass good 1's, so connect source to V_{DD}

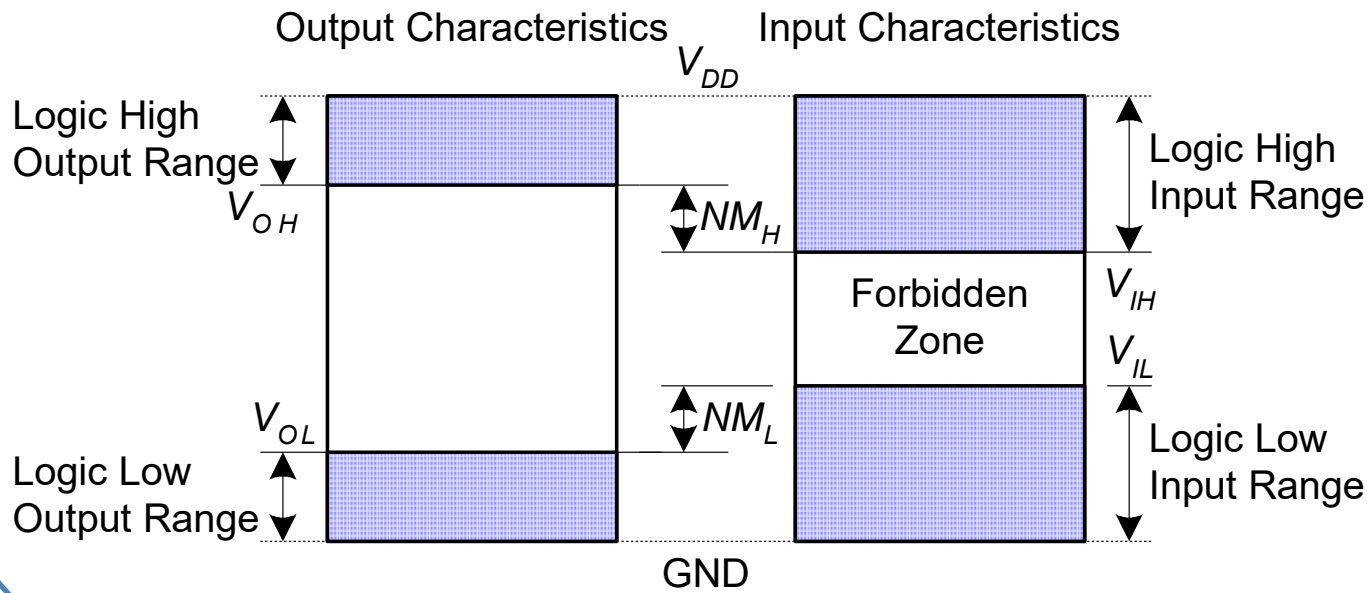
Binary Information Implemented with MOS transistors (Cont...)

A	p-type	n-type	$\sim A$
0 V	active	off	+3.3V
+3.3V	off	active	0V



A	$\sim A$
0	1
1	0

Binary Information Implemented with MOS transistors (Cont...)

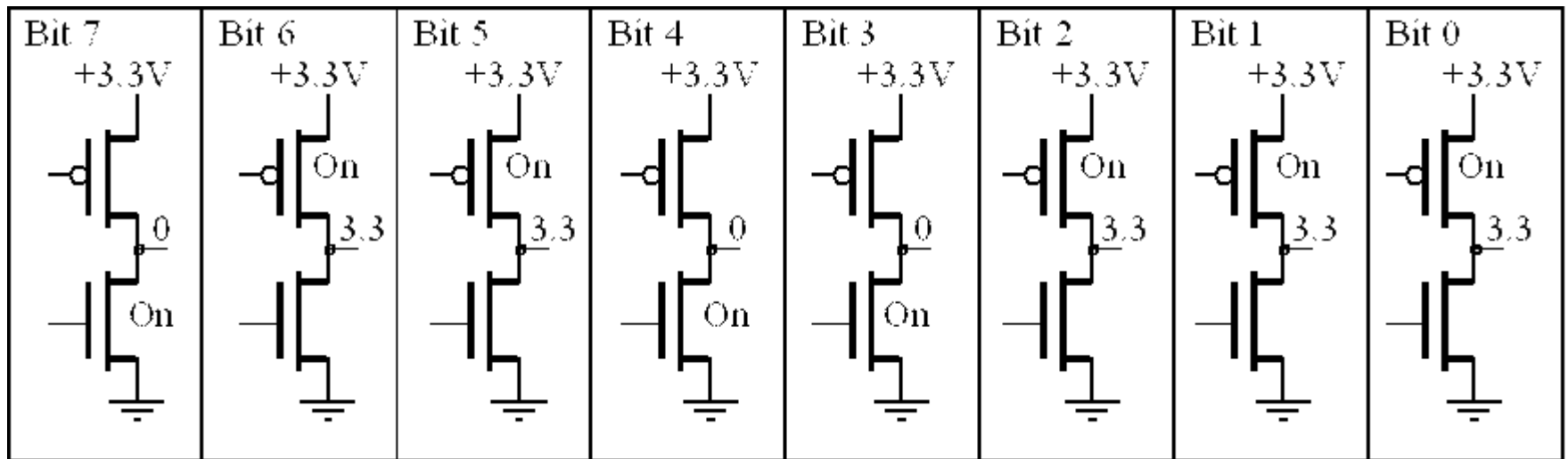


Binary Information Implemented with MOS transistors (Cont...)



Logic Family	V_{DD}	V_{IL}	V_{IH}	V_{OL}	V_{OH}
TTL	5 (4.75 - 5.25)	0.8	2.0	0.4	2.4
CMOS	5 (4.5 - 6)	1.35	3.15	0.33	3.84
LVTTL	3.3 (3 - 3.6)	0.8	2.0	0.4	2.4
LVCMOS	3.3 (3 - 3.6)	0.9	1.8	0.36	2.7

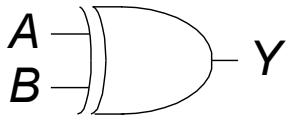
Binary Information Implemented with MOS transistors (Cont...)



A byte is comprised of 8 bits,
voltage **3.3V** means **true or 1** ; voltage of **0V** means **false or 0**.
In this case representing the binary number **01100111**.

Binary Information Implemented with MOS transistors (Cont...)

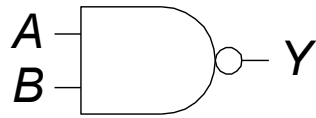
XOR



$$Y = A \oplus B$$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

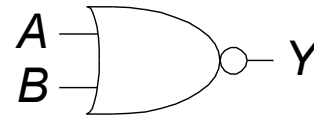
NAND



$$Y = \overline{AB}$$

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

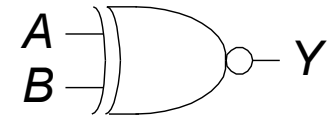
NOR



$$Y = \overline{A + B}$$

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

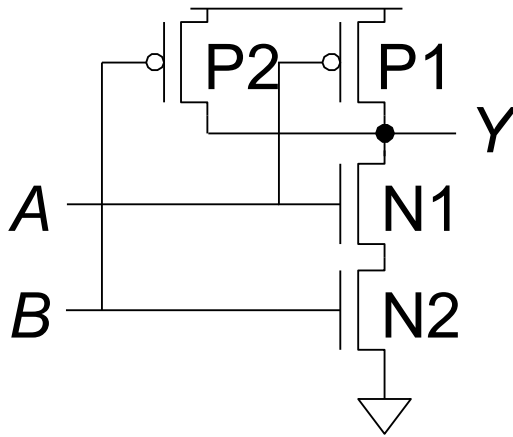
XNOR



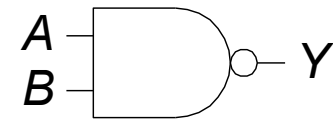
$$Y = \overline{A \oplus B}$$

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Binary Information Implemented with MOS transistors (Cont...)



NAND

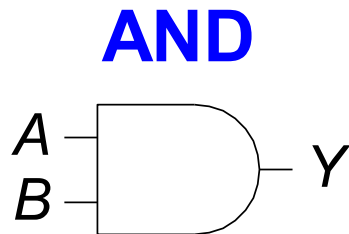


$$Y = \overline{AB}$$

A	B	P1	P2	N1	N2	Y
0	0	ON	ON	OFF	OFF	1
0	1	ON	OFF	OFF	ON	1
1	0	OFF	ON	ON	OFF	1
1	1	OFF	OFF	ON	ON	0

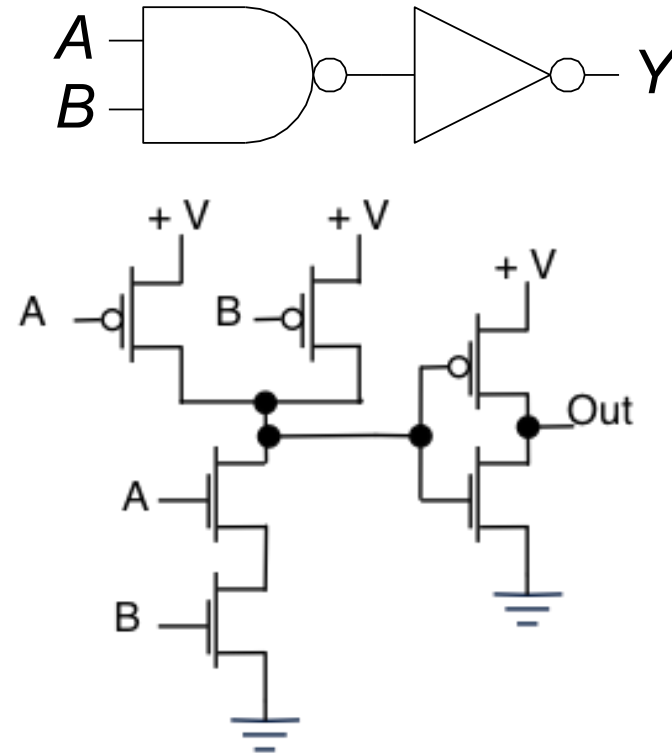
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Binary Information Implemented with MOS transistors (Cont...)



$$Y = AB$$

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



Assignment no. 2

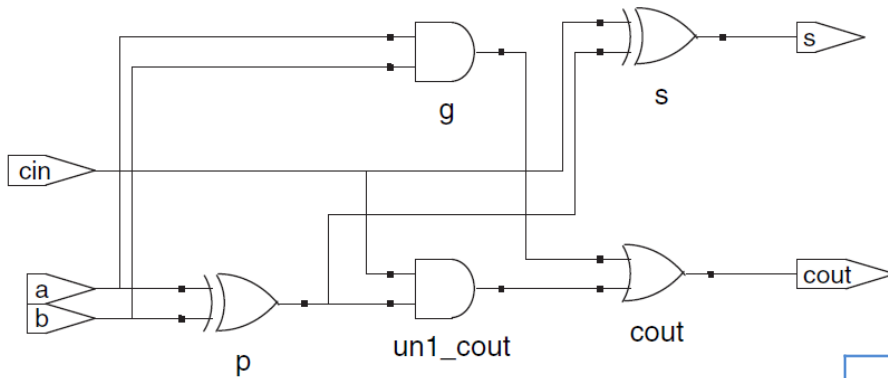
In the Banha-bank branch, It is proposed to install an embedded system to monitor the client queue in front of the tellers.

One of the requirements, the system should indicate the availability of the tellers in order to allow the clients to enter the queue. The manager said “If there are no tellers the system shouldn't allow clients entrance and a red indicator alarm is ON. But if there is one teller or more, the system should accept the clients and the alarm is OFF, given that there are four tellers”. Your team leader asks you to provide the circuit to implement the required function.

Embedded Systems

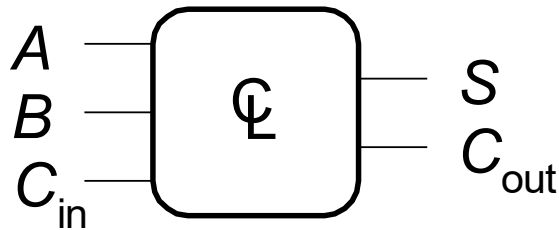
- ~~Introduction to Embedded System.~~
- ~~Review of Electronics Part (KCL, KVL, Parallel/Series Resistance, DAC and ADC).~~
- Review for Digital Systems (Binary Number, Logic, MOS Implantation, Computer Architecture).
- Intro to Programming Concepts (Structure and Concurrent).

Building Blocks (Adder)



$$S = F(A, B, C_{in})$$

$$C_{out} = F(A, B, C_{in})$$

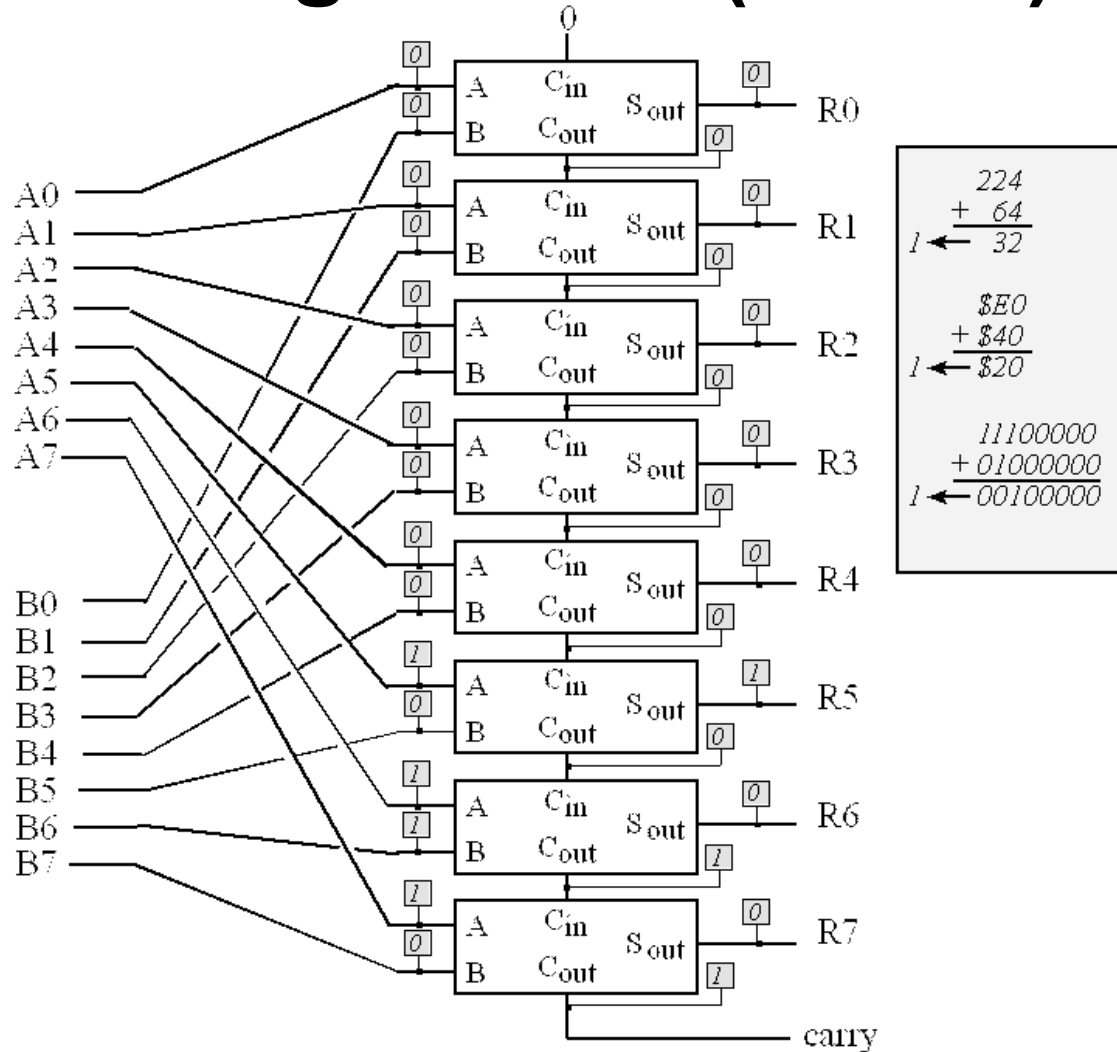


$$S = A \oplus B \oplus C_{in}$$

$$C_{out} = AB + AC_{in} + BC_{in}$$

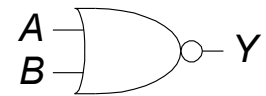
A	B	C_{in}	$A+B+C_{in}$	C_{out}	S_{out}
0	0	0	0	0	0
0	0	1	1	0	1
0	1	0	1	0	1
0	1	1	2	1	0
1	0	0	1	0	1
1	0	1	2	1	0
1	1	0	2	1	0
1	1	1	3	1	1

Building Blocks (Adder) cont...



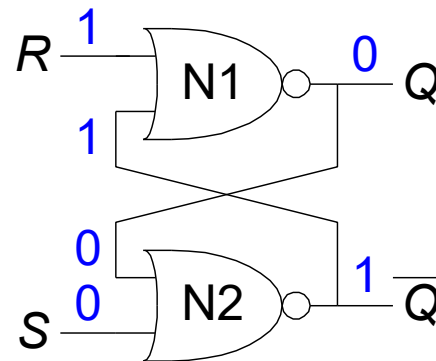
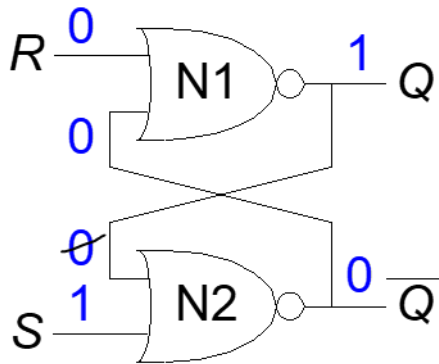
Storage Elements

- Digital storage elements are essential components used to **make registers and memory**.
- The simplest storage element is the **set-reset latch**.
 - $S = 1, R = 0$: **Set the output**
 - $S = 0, R = 1$: **Reset the output**
 - $S = 0, R = 0$: then $Q = Q_{prev}$ **Memory!**
 - $S = 1, R = 1$: **Invalid State** $Q \neq \text{NOT } Q$



$$Y = \overline{A + B}$$

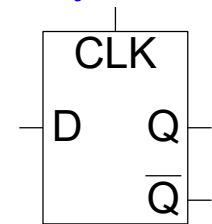
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0



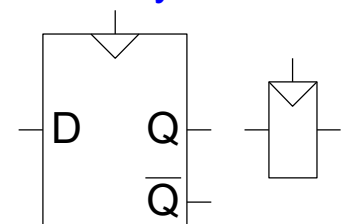
Storage Elements (Cont...)

- **D flip-flops** are the basic building **block of RAM and registers** on the computer.
- **D-Latch** Two inputs: *CLK*, *D*
 - **CLK**: controls *when* the output changes
 - **D** (the data input): controls *what* the output changes to
 - When **CLK** = 1, *D* passes through to *Q* (*transparent*)
 - When **CLK** = 0, *Q* holds its previous value (*opaque*)
 - Avoids invalid case when $Q \neq \text{NOT } Q$
- **D- flip flop** Two inputs: *CLK*, *D*
 - *Samples D on rising edge of CLK*
 - *When CLK rises from 0 to 1, D passes through to Q*
 - *Otherwise, Q holds its previous value*
 - *Q changes only on rising edge of CLK, Called edge-triggered*

D Latch
Symbol

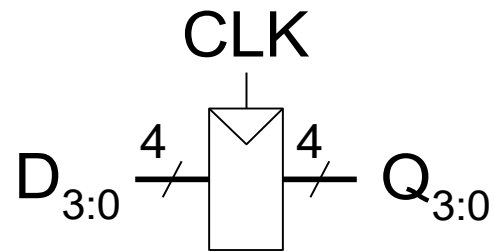
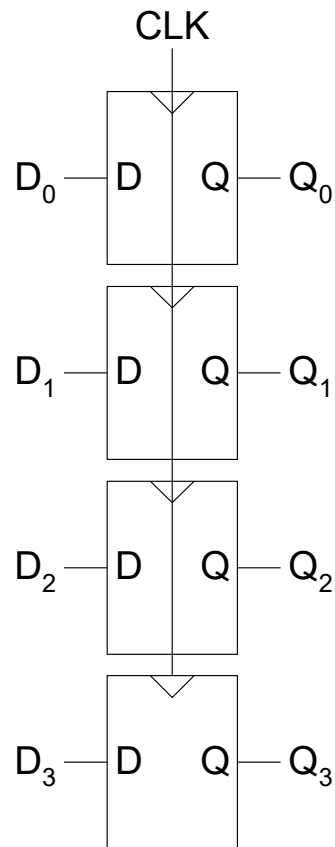


D Flip-Flop
Symbols



Storage Elements (Cont...)

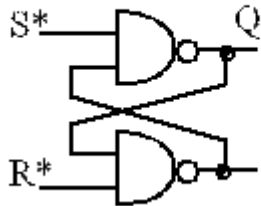
- **D flip-flops** are the basic building **block of RAM and registers** on the computer.



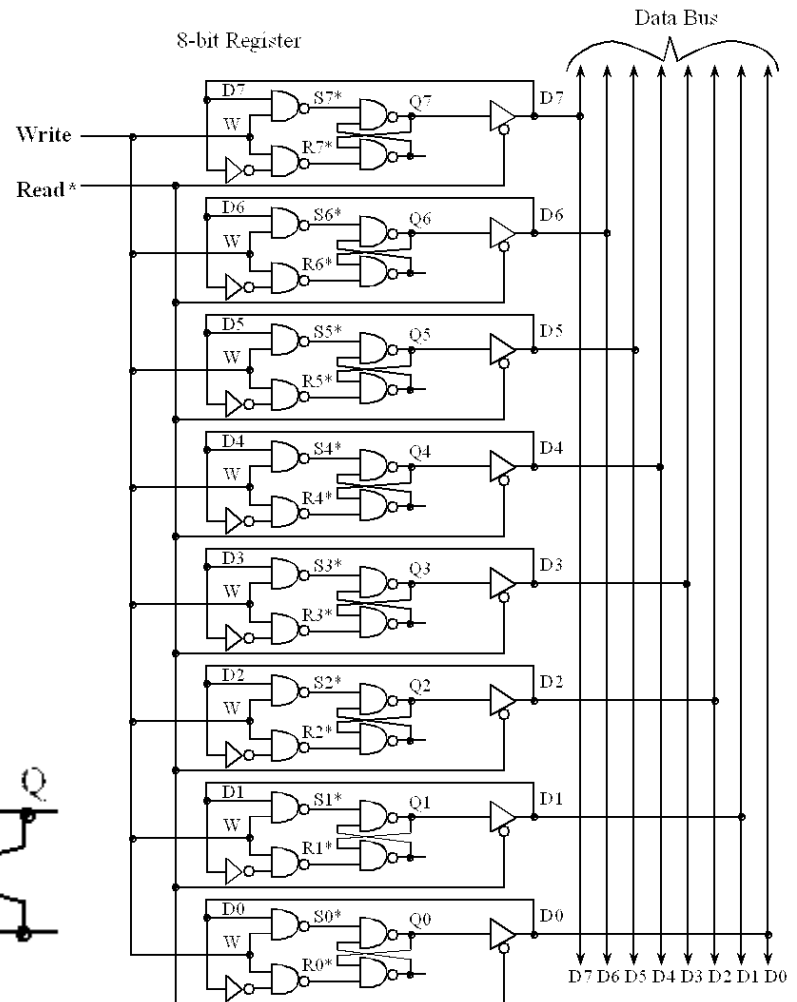
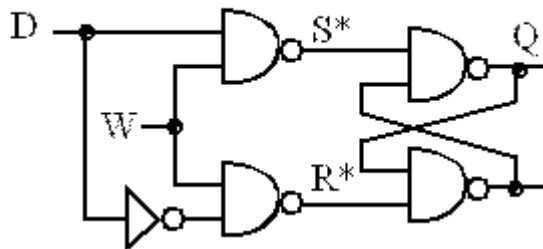
Storage Elements (Cont...)

- **D flip-flop** is the basic building block of RAM and registers on the computer.
- This basic storage element is called a **register**, as shown in Figure. (assembly).
- A **bus** is a collection of wires used to pass data from one place to another.

Set-Reset Latch

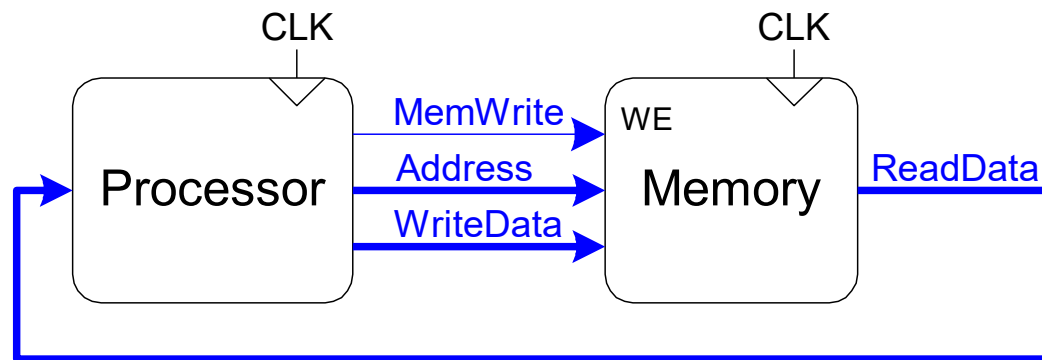


Gated D Latch



Digital Information stored in Memory

- Memory is a **collection** of hardware elements
- Each **memory cell** contains one **byte** of information, and each byte has a **unique** and sequential **address**. (byte-addressable)
- The **address** of a memory cell specifies its **physical location**.
- When we **write** to memory, we specify an **address** and 8, 16, or 32 bits of **data**, causing that information to be **stored** into the memory.
- When we **read** from memory we specify an **address**, causing 8, 16, or 32 bits of data to be **retrieved** from the memory.



Digital Information stored in Memory (Cont...)

- Software-Program is an **ordered sequence of very specific instructions** stored in memory, defining exactly **what and when certain tasks** are to be performed.
- The computer can store information in RAM by **writing** to it, or it can retrieve previously stored data by **reading** from it.
- Most microcontrollers have **static RAM (SRAM)** using **six metal-oxide-semiconductor** field-effect transistors to create each **memory bit**.
- **Flash ROM** is a popular type of **EEPROM**. Each flash bit **requires only two MOSFET transistors**. The input (gate) of one transistor is **electrically isolated**, charge is trapped on this input. The other transistor is used to read the bit by sensing whether or not the other transistor has trapped charge.
- Because flash is smaller than regular EEPROM, **most microcontrollers have a large flash** into which we store the software. For all the systems in this class, we will **store instructions and constants in flash ROM and place variables and temporary data in static RAM**.

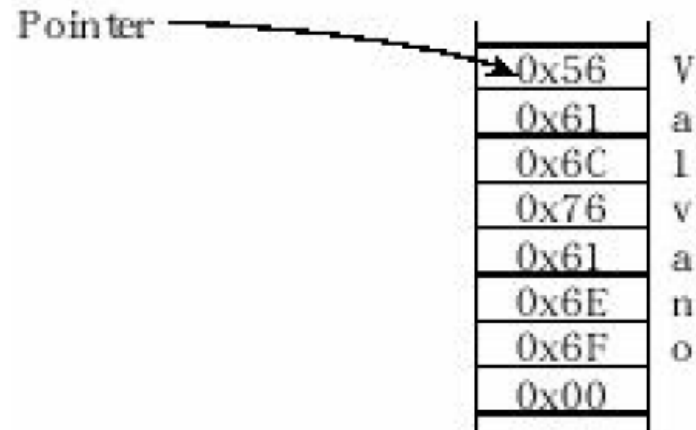
Digital Information stored in Memory (Cont...)

ROM	RAM
The information is programmed or burned into the device, and during normal operation it only allows read accesses.	The information is stored temporary , and during normal operation we can read from or write data into RAM.
nonvolatile , meaning the contents are not lost when power is removed.	volatile , meaning the contents are lost when power is removed.
ROM on the other hand is much denser than RAM.	Most microcontrollers have much more ROM than RAM.
It takes a comparatively long time to program or burn data into a ROM. (1ms)	Writing to RAM is about 100,000 times faster (on the order of 10 ns).

Character information

- American Standard **Code** for Information Interchange (**ASCII**) code is used to represent a **character**.
- Standard ASCII is actually only **7 bits**, but is stored **8-bit** bytes with the most significant bit equal to 0.
- For example, the capital ‘V’ is defined by the 8-bit **binary** 0101_0110 **hexadecimal** 0x56.
- In C, the **char data type** is used to represent **characters**.

- “Valvano” is encoded as these 8 bytes
0x56, 0x61, 0x6C, 0x76, 0x61, 0x6E,
0x6F, 0x00 (**NULL character**).



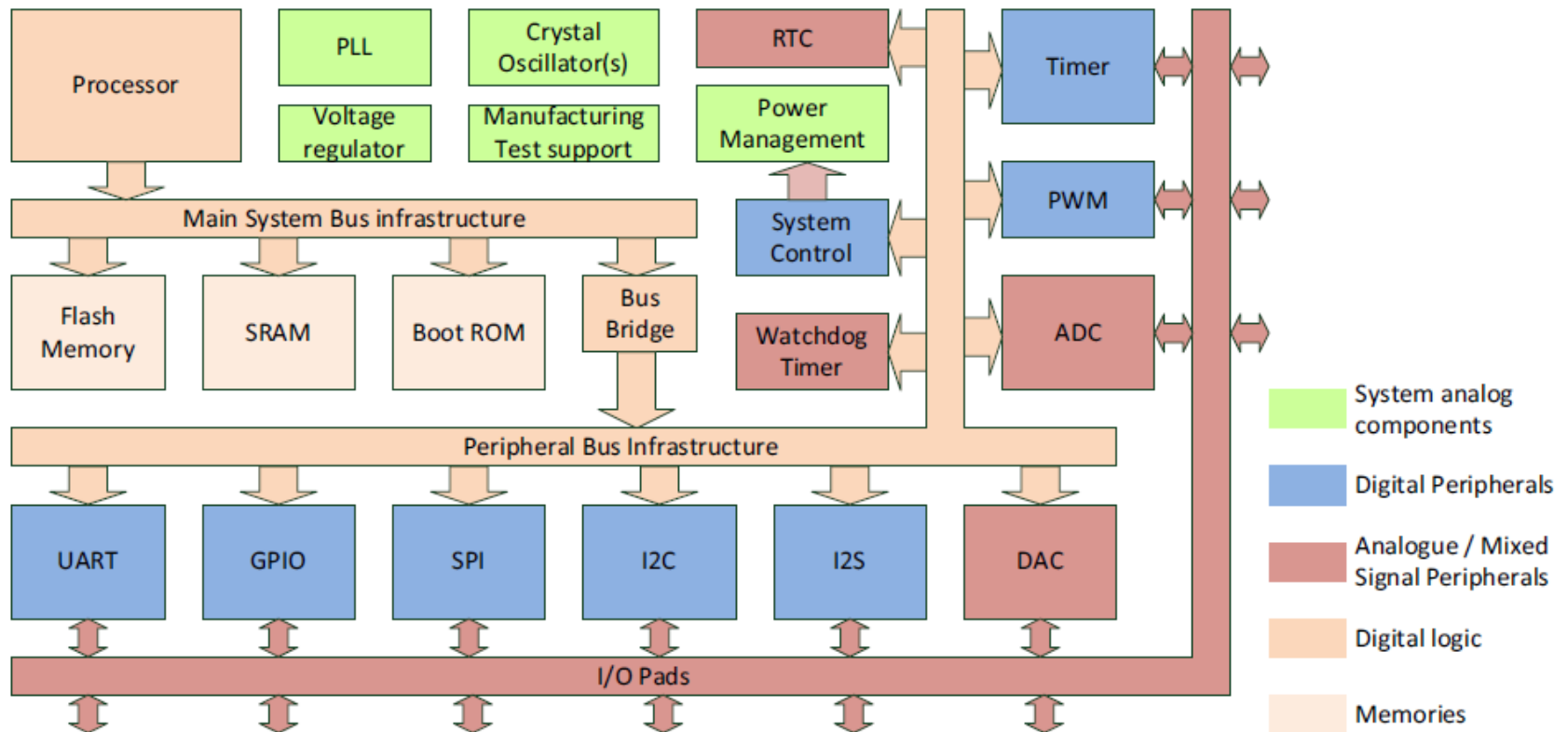
HW- Computer Architecture

- A computer combines a processor, random access memory (RAM), read only memory (ROM), and input/output (I/O) ports.
- Computers are **electronic idiots**. They can **store** a lot of data, **execute** programs quite quickly but they **do exactly what we tell them to do**. They don't get bored doing the same tasks over and over again.
- A **micro-computer** is a small computer, where small refers to **size**.
- A very small micro-computer, called a **microcontroller**, contains all the components of a computer (processor, memory, I/O) on a single chip.
- A **port** is a **physical connection** between the computer and its outside world. Information **enters** via the **input ports** and **exits** via the **output ports**.
- A **bus** is a **collection of wires** used to **pass** information between modules.

HW- Computer Architecture (Cont...)

- An **interface** is defined as the collection of the I/O port, external electronics, physical devices, and the software, which combine to allow the computer to **communicate** with the external world.
- An example of an input interface is a **switch**, where the operator toggles the switch, and the software can recognize the switch position. An example of an **output interface** is a light-emitting diode (LED), where the software can turn the light on and off.
- In general, we can classify I/O interfaces into four categories
 - **Parallel**- binary data are available **simultaneously** on a group of lines.
 - **Serial**- binary data are available **one bit at a time** on a single line.
 - **Analog**- data are **encoded as an electrical voltage, current, or power**.
 - **Time**- data are encoded as a **period, frequency, pulse width, or phase shift**.

HW- Computer Architecture (Cont...)



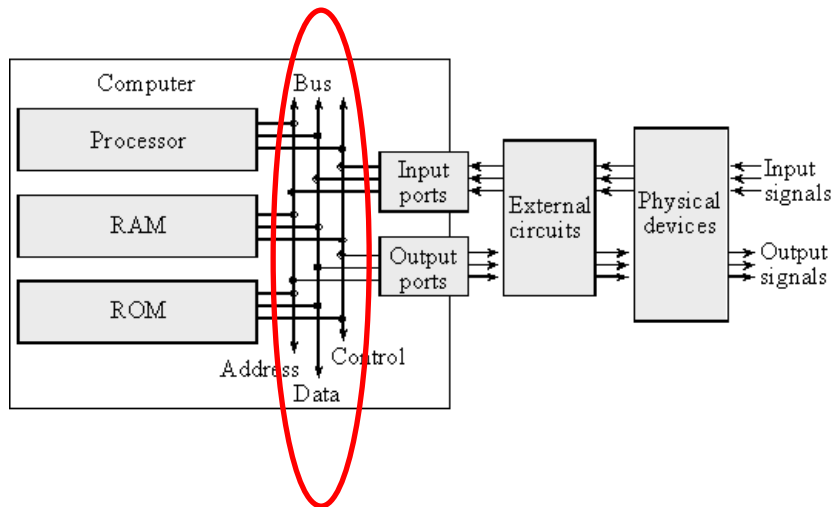
A simple microcontroller.

HW- Computer Architecture (Cont...)

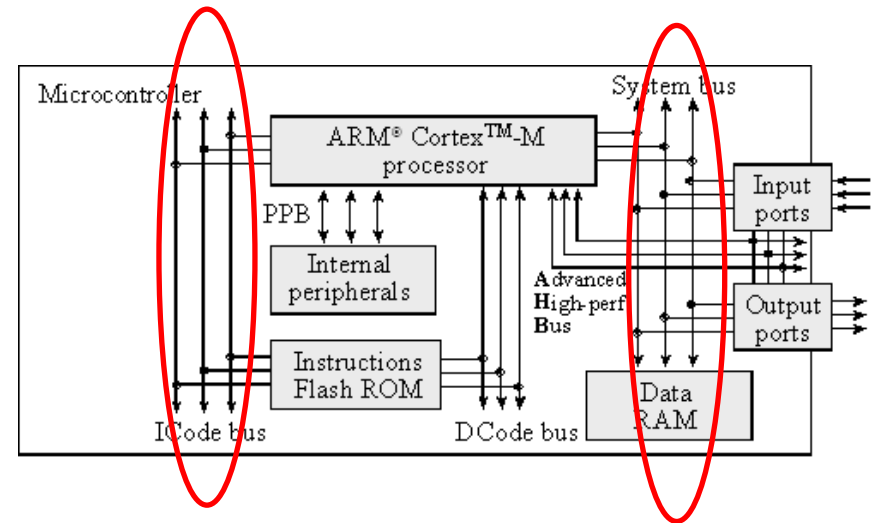
Item	Descriptions
ROM	Read Only Memory—Nonvolatile memory storage for program code.
Flash memory	A special type of ROM, which can be reprogrammed many times, typically for storing program code.
SRAM	Static Random Access Memory—for data storage (volatile)
PLL	Phase Lock Loop—a device to generate programmable clock frequency based on a reference clock.
RTC	Real Time Clock—a low power timer for counting seconds (typically runs on a low power oscillator), and in some cases also for minutes, hours and calendar functions.
GPIO	General Purpose Input/Output—a peripheral with parallel data interface to control external devices and to read back external signals status.
UART	Universal Asynchronous Receiver/Transmitter—a peripheral to handle data transfers in a simple serial data protocol.
I2C	Inter-Integrated Circuit—a peripheral to handle data transfers in a serial data protocol. Unlike UART, a clock signal is required and can provide higher data rate.
SPI	Serial Peripheral Interface—another serial communication interface for off-chip peripherals.
I2S	Inter-IC Sound—a serial data communication interface specifically for audio information.
PWM	Pulse Width Modulator—a peripheral to output waveform with programmable duty cycle.
ADC	Analog to Digital Converter—a peripheral to convert analog signal-level information into digital form.
DAC	Digital to Analog Converter—a peripheral to convert data values into analog signal level.
Watchdog timer	A programmable timer device for ensuring the processor is running program. When enabled, the program running needs to update the watchdog timer within a certain time gap. If the program crashed, the watchdog timed out and this can be used to trigger a reset or a critical interrupt event.

HW- Computer Architecture (Cont...)

- Von Neumann architecture

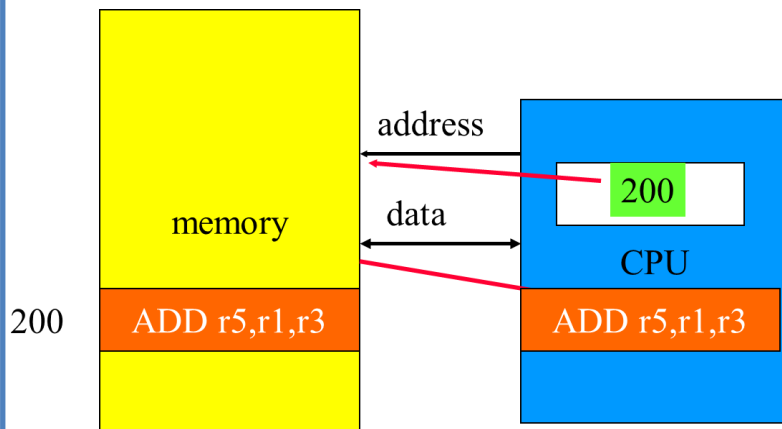


- Harvard architecture

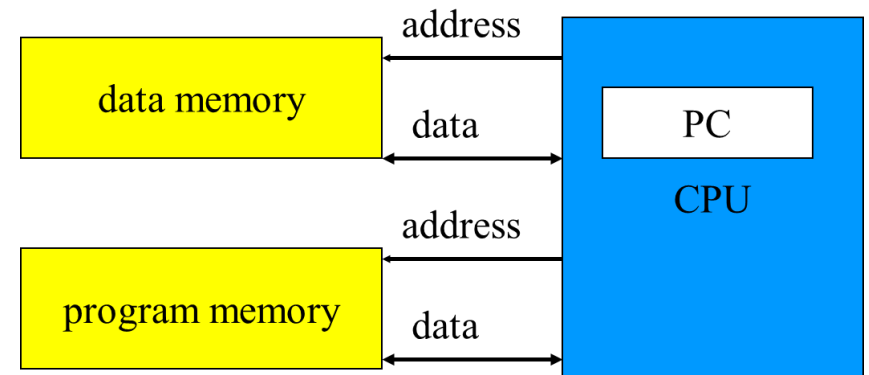


HW- Computer Architecture (Cont...)

- Von Neumann architecture



- Harvard architecture

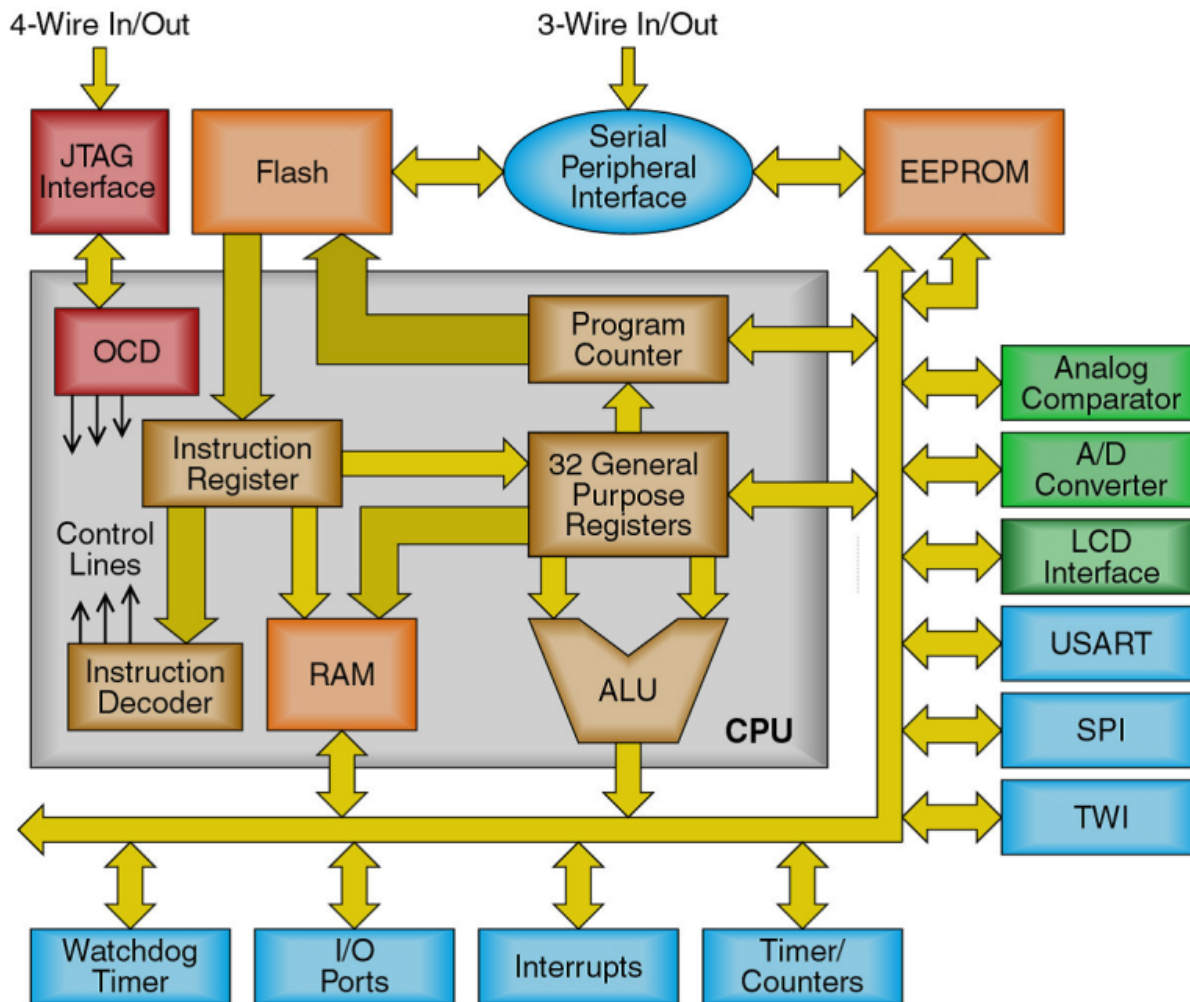


HW- Computer Architecture (Cont...)

RISC vs. CISC

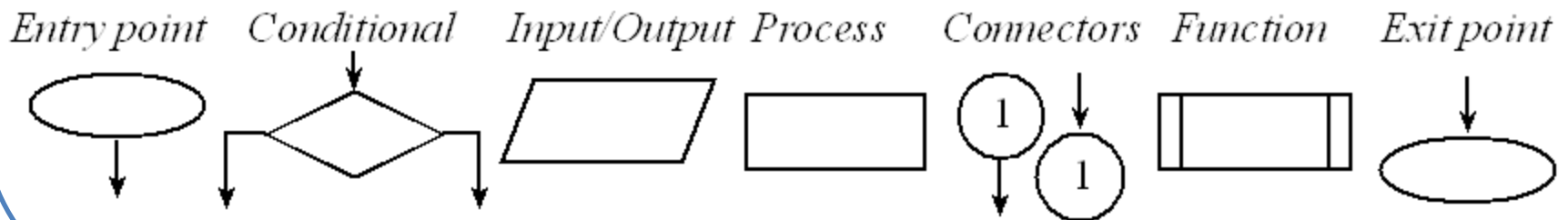
- ❑ **Complex** instruction set computer (CISC):
 - many addressing modes;
 - many operations.
- ❑ **Reduced instruction** set computer (RISC):
 - load/store;
 - pipelinable instructions.

HW- Computer Architecture (Cont...)



SW- Structured Programming

- graphical tools are used to describe the organization of an embedded system such as: **flowcharts**, **data flow graphs**, and **call graphs**.
- Programs are written in a **linear** or **sequential/one-dimensional** fashion.
- **Conditional branching** and **function calls** create complex behaviors that are not easily observed in a linear fashion.
- Flowcharts are one way to describe software in a **two-dimensional format**, to visualize conditional branching and function calls.
- Flowcharts are very useful in **the initial design stage** and it used in **the final documentation** stage of a project.

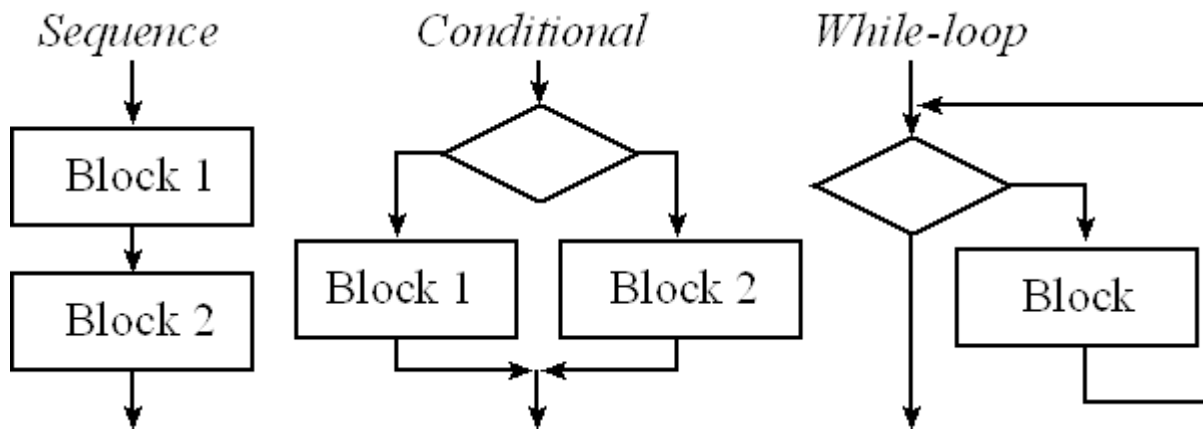


SW- Structured Programming

- The **entry point** is the **starting point** of the software/ function, or subroutine. The **exit point** returns the flow of control back to the place from which the function was called.
- The **Rectangles** specifies the **process** block. In a high-level flowchart, a process block might involve many operations.
- The **parallelogram** defines an **input/output operation**.
- The **diamond-shaped** defines a **branch point** or **conditional block**. Each arrow out of a condition block must be labeled with the condition causing flow to go in that direction. The condition for each arrow must be **mutually exclusive**.
- The **Rectangle with double lines** on the side specifies a **call to a predefined function**.
- **Circles** are used as **connectors**. Connectors with an arrow pointing into the circle are **jumps** or **goto commands**.

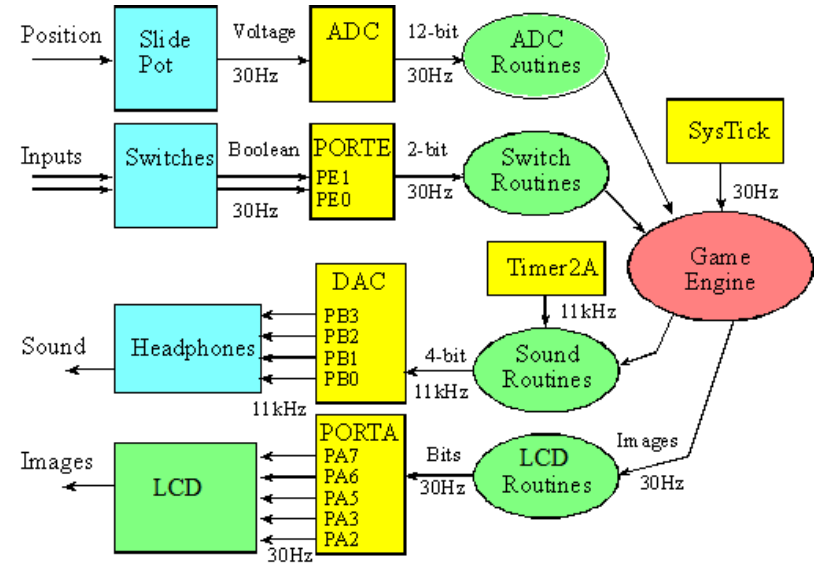
SW- Structured Programming

- **Structured programs** are built from three basic building blocks: the **sequence**, the **conditional**, and the **while-loop**.
- At the lowest level, the process block contains simple and well-defined commands.

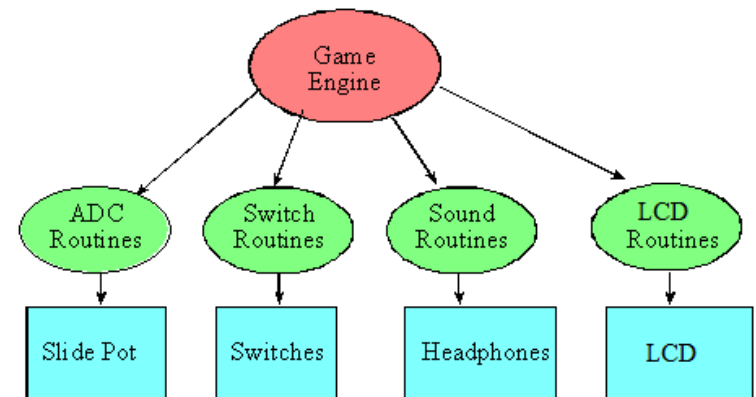


SW- Structured Programming

- **Data flow graph** is a block diagram of the system, showing **the flow of information**. Arrows point **from source to destination**.








- **Call graph** is a graphical way to define how the software/hardware modules interconnect.



SW- Parallel Programming

- **Parallel programming** to execute **multiple threads** at the same time. A computer with a **multi-core processor** can **simultaneously execute** a separate program in each of its cores.
- Fork and join are the fundamental building blocks of parallel programming.
- After a **fork**, two or more software threads run in **parallel/ simultaneously** on separate processors. Two or more simultaneous software threads can be **combined** into one using a **join**.
- **Concurrent programming** allows the computer to execute **multiple threads, but only one at a time**.
- **Interrupts** are one mechanism to **implement concurrency** on real-time systems. The **foreground** thread is defined as the execution of the **main** program, and the **background** threads are executions of the **Interrupt**.

Lab

Remove Product(s)		Qty.	Total
<input type="checkbox"/>	 Led Matrix 5x7 - Red	<input type="text" value="1"/>	10.00L.E.
<input type="checkbox"/>	 Accelerometer (Triple Axis) Module - Digital - ADXL345	<input type="text" value="1"/>	60.00L.E.
<input type="checkbox"/>	 Infrared Receiver Module "KSM803"	<input type="text" value="1"/>	7.50L.E.
<input type="checkbox"/>	 Temperature & Humidity Sensor (DHT-11) Module "KY-015"	<input type="text" value="1"/>	35.00L.E.
<input type="checkbox"/>	 Ultrasonic Sensor & Distance Measurement Module HC-SR04	<input type="text" value="1"/>	48.00L.E.

NCS 362: Embedded Systems

- ~~Intro to Embedded System.~~
- ~~Review of Electronics Part (KCL, KVL, Parallel/Series Resistance, DAC and ADC).~~
- ~~Review for Digital Systems (Binary Number, Logic, MOS Implantation, Computer Architecture).~~
- ~~Intro to Programming Concepts (Structure and Concurrent).~~