EC354: Embedded and Real Time Operating systems (3-0-0)

Department of Electronics and Communications, NIT Warangal

2. Embedded Systems Architecture - 1

Teja Mannepalli

December 31, 2025

1 Embedded systems - architecture

The typical embedded system has the following components as shown in Fig. 2.3.2. The embedded systems are designed to control, regulate or manipulate a physical variable in any industry. The embedded systems send signals to actuators in response to the inputs provided by the sensors.

1. *System core* is the brain of the embedded system that takes input from the sensor and interacts with memory to decide what to do in real-time. The system core is typically a microcontroller, microprocessor, FPGA, or ASIC. This is the master brain of the entire system. The system core is in general firmware and does not permit any changes by the end user.

2. *Sensors and Actuators*: The system core receives sensor signals about a physical variable. It interacts with the memory and takes a decision and signals are sent to the actuator. The actuator amplifies the output signals from embedded systems and controls the physical variable in the industry or plant.

3. *Memory* holds the control algorithm and other instructions needed. ROM (Read-only memory) stores the program code and retains the memory even if the power is off. ROM generally does not allow the user to modify the code. Ex: OTP, PROM, EPROM, Flash, etc. RAM (Random Access Memory) is often the 'working' memory of the processor/ controller. ex: SRAM, DRAM, etc. RAM is volatile, meaning the contents are erased when the power is off.

4. *Communication Interfaces* The Embedded systems have different interfaces that help the processor to communicate either within the internal components of the embedded system or with the outside world.

2 CPU or system core

The central processing unit (CPU) often refers to any of the two categories either a general-purpose processor or a digital signal processor. GPPS are either microcontrollers or processors. ALU performs arithmetic and logic operations. General Purpose registers contain processors' internal memory. They contain data and operands used by the processor. If the processor is 8 bits, it refers to GPRs being 8-bit. Instruction Pointer points to the next instruction to be executed. Also known as program counter. The stack pointer points to the stack in the memory. In external memory, the processor implements the stack. If needed, the contents of the registers are transferred to the stack. The instruction decoder decodes the instructions it gets.

A CPU is classified into the general purpose processor (GPP) and Digital Signal Processor (DSP) categories. GPPs are further classified as microcontrollers (small-sized independent embedded systems) and microprocessors (more powerful processing but need external components). ALU is the Arithmetic Logic Unit that performs arithmetic and logic operations. General Purpose registers contain processors' internal memory. Registers contain data and operands that the processor uses.

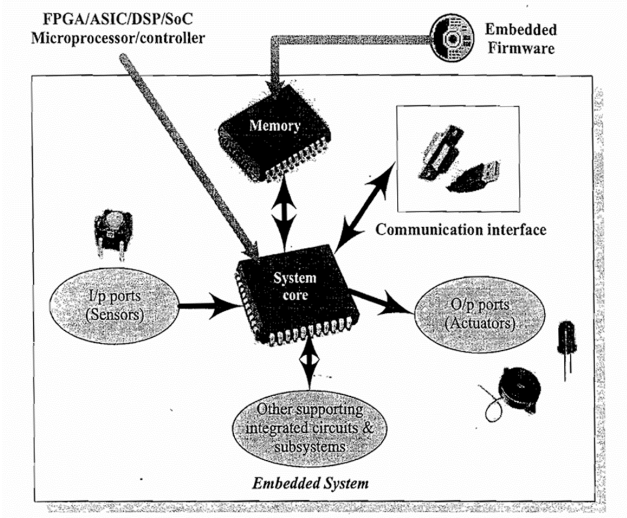

Figure 1: Architecture of a typical embedded system.

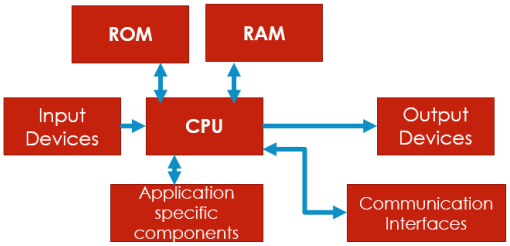

Figure 2: Architecture of a CPU.

Figure 3: CPU and memory communication channels.

2.1 Microprocessor

A microprocessor is a CPU capable of performing arithmetic and logic operations according to a pre-defined set of instructions. It contains an Arithmetic Logic Unit (ALU), a control unit and working registrars tightly linked with a memory. Today 2.4 GHz microprocessors are available in the market. Unlike microcontrollers, it is a dependent unit that often needs external hardware components like memory, timers, and interrupt controllers. The components of a microprocessor are given in Fig. 4. The general purpose registers constitute the processor's internal memory. These have current data and operands that the processors are manipulating. The control unit fetches the instructions from memory, decodes them and executes them. The instruction pointer (or program counter) points to the next instruction that needs to be executed. The stack pointer points to the stack in the memory. When required the contents of the stack are transferred to the stack. The processor keeps track of the next free location in the stack through the stack pointer. A stack pointer is a register that stores the memory address of the last data element added to the stack.

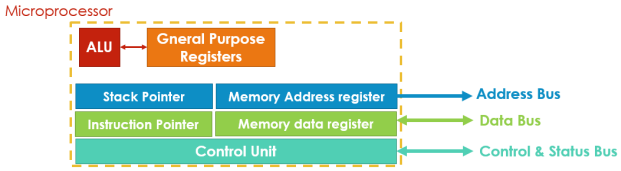

Figure 4: Architecture of a typical microprocessor

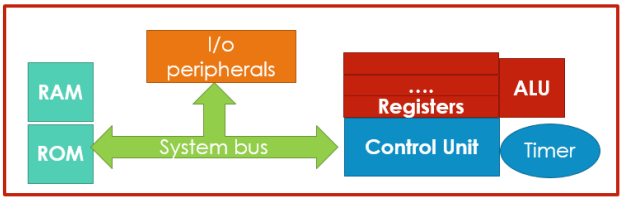

Figure 5: Architecture of a typical microcontroller

The processor also is responsible for reading/ writing the data to the memory, input devices or output devices. The processor also needs to communicate with other components using buses. As it is shown in Fig. 2.3.2, there are three busses used by the μP to communicate with other devices.

1. Data bus carries the data from the processor to other devices (bi-directional).
2. Address bus carries the address information from the processor to memory (bi-directional)
3. Control and status bar carries control/ status information. Examples are: the status of an operation going on, processor reset signal, clock input and interrupt signal, etc.

2.2 Microcontroller

A microcontroller is a highly integrated chip that contains CPU, RAM, special, general purpose registers, on-chip ROM/ FLASH memory for program storage, a timer, and interrupt controller, and dedicated I/O ports. μC's are the superset of μP's. These are self-contained units and do not require any external interrupt controllers, timers, or UART for functioning. There are multiple I/O ports which can be either 8-bit, 16-bit or 32-bit ports.

Microcontrollers usually must have low-power requirements since many devices they control are battery-operated. Microcontrollers are used in many consumer electronics, car engines, computer peripherals, and test or measurement equipment, and are well-suited for long-lasting battery applications. Microcontrollers are designed for embedded applications, in contrast to the microprocessors used in personal computers.

In an 8-bit4 microcontroller, the arithmetic logic unit performs the arithmetic and logic operations. An 8-bit integer can hold a value up to 255. Examples of 8-bit microcontrollers include Intel 8031/8051, PIC1x, and Motorola MC68HC11 families.

A 16-bit microcontroller performs greater precision and performance compared to an 8-bit microcontroller. A 16-bit integer can hold a value up to 65,535 and it can automatically operate on two 16-bit numbers. Some examples of 16-bit microcontrollers are 16-bit Intel 8096 and Motorola MC68HC12 families.

The 32, 64, and 128-bit microcontrollers use much larger values to perform the arithmetic and logic operations. These are used in automatically controlled devices including implantable medical devices, engine control systems, office machines, appliances, and other types of embedded systems. Some examples are Intel/Atmel 251 family and PIC3x.

2.3 von-Neumann v/s Harvard Architecture

2.3.1 von-Neumann Architecture

Von-Neumann architecture: has a single common bus for fetching both instructions and memory. Instructions and data are stored in main memory. The processor/ controller searches for the instruction followed by data that supports the instruction from the memory.

2.3.2 Harvard Architecture

Harvard architecture: has separate instruction bus and data bus. This allows data transfer and program fetching simultaneously. The program memory can be read/ written with access to program memory. The separate things allow to execute an instruction and the next instruction is fetched. The figure shows a schematic of both architectures.

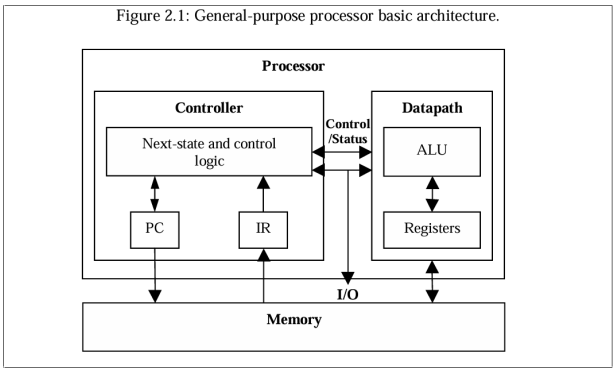

Figure 6: General-purpose processor basic architecture in a different way. Try to correlate with Fig. 4.

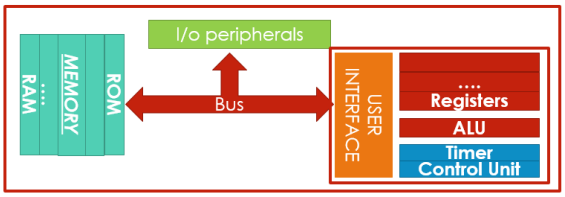

Figure 7: von-Neumann Architecture.

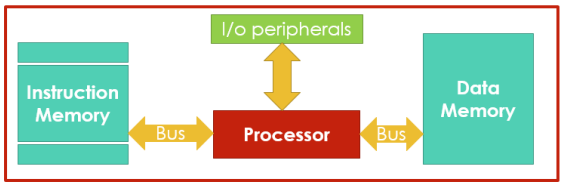

Figure 8: Harvard Architecture

2.4 Endianness

Each byte in the memory has a unique address. Let us consider a 4-bit memory. Each of the 4-bit memory has an address. Endianess stores the order in which data is stored in the memory. Let us say, a 4 byte long integer is in the memory. According to the Little Endian, the lowest order byte is stored in the lowest is stored in the lowest address, and highest order byte is stored in the highest address.

This is depicted in Figure 9.

3 Memory

The on-chip memory refers to inbuilt memory in the processors/ controllers. The off-chip memory refers to external memory to be connected to processors/ controllers. Masked ROM is a one-time programmable device. This is programmed in the factory itself by the metallization process at production. Least expensive

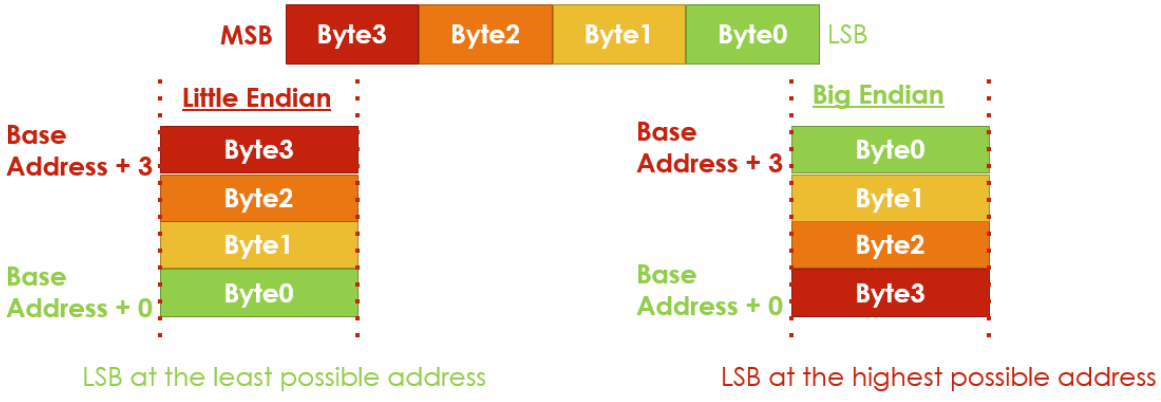

Figure 9: Little v/s big Endian.

type of solid-state memory. A programmable read-only memory (PROM) is a form of digital memory where the contents can be changed once (by the user and not by the manufacturer) after the manufacture of the device. The data is then permanent and cannot be changed. It has nichrome or polysilicon wires in a matrix. An EPROM (kind of EROM), or erasable programmable read-only memory, retains its data when its power supply is switched off. EPROM stores a bit by charging the floating gate of FET. It needs to be taken out of the chip and put in a UV eraser device. An electrically erasable programmable read-only memory uses electrical signals to store bits.

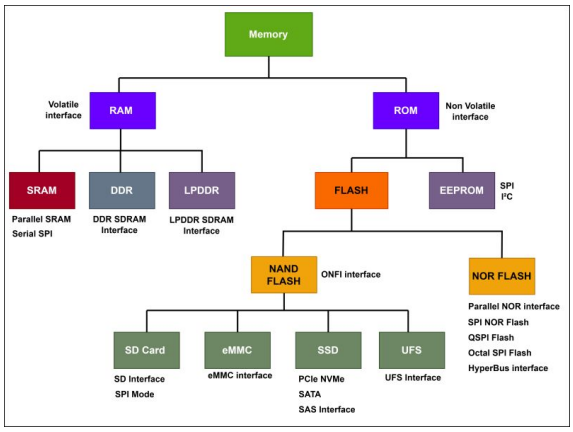

Figure 10: Classification of memory.

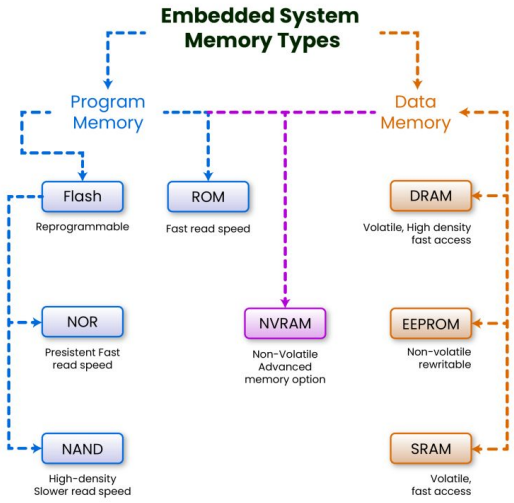

Figure 11: Program and Data memory storage.

4 Read-only memory (ROM)

Read-only memory (ROM) is a type of non-volatile memory used in computers and other electronic devices. Data stored in ROM cannot be electronically modified after the manufacture of the memory device. Read-only memory is useful for storing software that is rarely changed during the life of the system, also known as firmware.

4.1 Key points about ROM:

Unlike RAM, data in ROM persists even when power is lost. Information stored in ROM can only be read, not modified. Storing essential device instructions, firmware, and unchanging data. ROM can hold the data when the power supply is OFF too. The ROM is different from RAM which is volatile (the data will be erased when there is no power supply.

4.2 Examples of ROM devices:

Masked ROM is a one-time programmable device. This is programmed in the factory itself by the metallization process at production. Least expensive type of solid-state memory. *programmable read-only memory (PROM)* is a form of digital memory where the contents can be changed once (by the user and not by the manufacturer) after the manufacture of the device. The data is then permanent and cannot be changed. It has nichrome or polysilicon wires in a matrix. *EPROM (rarely EROM)*, or erasable programmable read-only memory, retains its data when its power supply is switched off. EPROM stores a bit by charging the floating gate of FET. It needs to be taken out of the chip and put in a UV eraser device. An electrically erasable programmable read-only memory uses electrical signals to store bits. *FLASH memory* is the latest and most

popular ROM that stores the bits in an array of floating gate MOSFETS and is organized as sectors (blocks) or pages.

4.3 Operation of a Flash (ROM) memory

FLASH memory is the latest and most popular ROM that stores the bits in an array of floating gate MOSFETS and is organized as sectors (blocks) or pages. The basic unit of Flash memory is the floating gate MOSFET. The schematic is shown below: (The difference from MOSFET is the Floating gate between the control and substrate. Rest is similar to conventional NMOS. Logic '1" is stored in FLASH, which means no electrons are trapped in the floating gate. If we apply the voltages shown, a conducting channel of current is between the source and the drain.

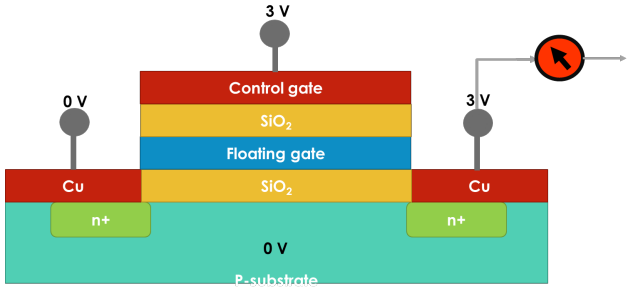

Figure 12: Architecture of a typical FLASH memory cell.

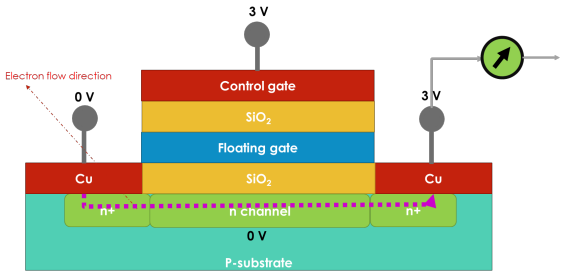

Figure 13: Reading logic 1 in FLASH memory.

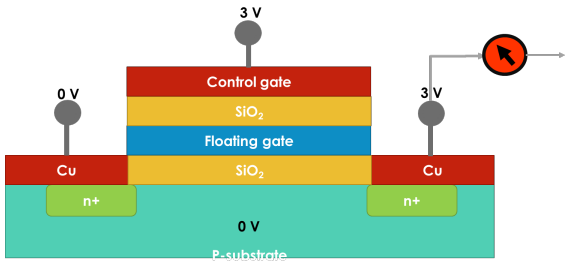

Figure 14: Reading logic 0 in FLASH memory.

Let us nomenclature of *Cnt-LeCu-pSub-RiCu* representing the Control Gate, Left Copper gate, Substrate and Right Copper gate.

Reading logic '1"

The voltages across the *Cnt-LeCu-pSub-RiCu* are applied as 3 V, 0 V, 0 V and 3 V. If let us say, a cell is uncharged, the electrons on its floating gate cannot interfere with the current flow, allowing a strong signal to be read as a '1".

Reading logic '0"

The voltages across the *Cnt-LeCu-pSub-RiCu* are applied as 3 V, 0 V, 0 V and 3 V. When a cell is charged, the electrons on the floating gate hinder the current flow, resulting in a weaker signal read as a "0".

Writing logic '1"

The voltages across the *Cnt-LeCu-pSub-RiCu* are applied as 10 V, 0 V, 0 V and 7 V. Writing logic '1' corresponds to the erase operation. During erasing, a high electric field is applied across the tunnel oxide by

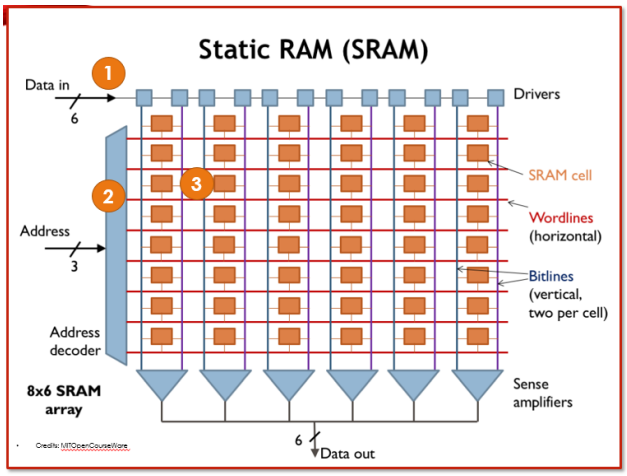

Figure 15: Caption

applying a high voltage to the control gate (or substrate, depending on the flash architecture). This strong electric field causes electrons stored on the floating gate to be removed via Fowler–Nordheim tunnelling. Erase operations are performed at the block level, setting all cells within the block to the erased state, i.e., logic '1'.

Writing logic "0"

The voltages across the *Cnt-LeCu-pSub-RiCu* are applied as 0 V, 10 V, 10 V and 10 V. Writing logic '0' corresponds to the program operation. During programming, electrons are injected into the floating gate using hot-electron injection or Fowler–Nordheim tunnelling, depending on the flash memory type. The injected electrons increase the threshold voltage of the transistor, preventing channel conduction during a subsequent read operation. Program operations are typically performed at the page level within a previously erased block.

1. Flash memory must be erased before programming; erasing sets all cells in a block to logic '1'.
2. Flash memory is erased in blocks, while programming is carried out in pages within a block.
3. Flash memory cells have a limited number of program/erase cycles due to tunnel oxide degradation.

5 Random Access Memory (RAM)

RAM is the working or data memory of the controller/ processor to read and write from it. RAM is a volatile memory meaning the contents are erased when the power is turned off. The desired memory location can be randomly accessed in contrast to Sequential Access Memory (SAM). In SAM, the entire memory has to be traversed to access a desired memory location. The CD-ROMs, magnetic tapes are the best examples of SAMs. RAM has three categories- static RAMs, dynamic RAMs and non-volatile RAMs. Generally, RAM is known to be volatile.

5.1 Static RAM

The figure depicts an 8 x 6 SRAM (Static Random Access Memory) array organized as a grid with 8 rows and 6 columns. Each row corresponds to a memory location, while each column represents a bit in a memory word. This configuration allows the SRAM to store 8 memory words, each consisting of 6 bits. Memory access involves either reading or writing all the bits of a single memory word at a time.

Addressing in the SRAM is accomplished using 3 address bits, which select one of the 8 memory locations. The 6-bit word from the selected location is output through the data-out port. For larger SRAM arrays,

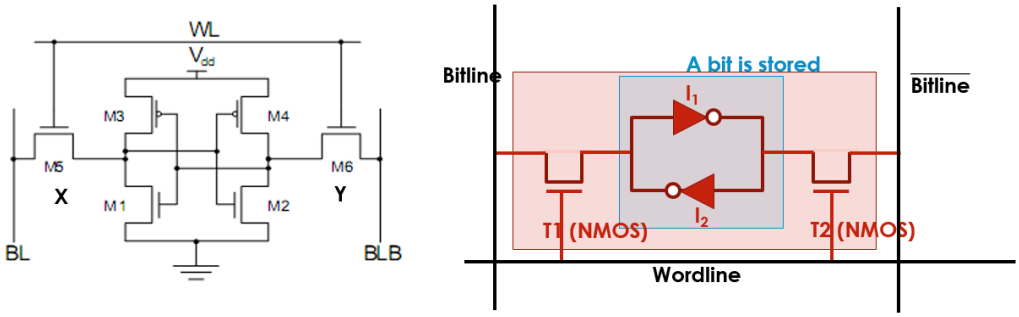

Figure 16: The 6T SRAM cell circuit is shown.

more sophisticated layouts are often used to minimize the length and capacitance of the bitlines, enhancing performance.

The SRAM cell itself is constructed using two CMOS inverters connected in a positive feedback loop, forming a bi-stable storage element. This design allows the cell to retain its stored value as long as power is supplied, even in the presence of some noise.

The operation of the SRAM is controlled by the wordlines and bitlines:

1. *Wordline*: This signal activates a specific row of memory cells, effectively selecting a memory address for access.

2. *Bitline*: This signal carries the data to or from the memory cells in the selected row, enabling read or write operations.

When the wordline connected to the gates of the access transistors (FETs) is high, the FETs turn on, establishing an electrical connection between the cell's internal circuitry and the bitlines. This allows data to be read from or written to the cell. Conversely, when the wordline is low, the access FETs turn off, isolating the bi-stable feedback loop from the bitlines and ensuring the stored value remains stable as long as power is supplied.

Circuit Description

The 6T SRAM cell consists of six MOSFETs labeled *M1–M6*: *M1 and M2:* These are the *pull-down transistors* in the cross-coupled inverter latch. They help maintain the stored value. *M3 and M4:* These are the *pull-up transistors* in the cross-coupled latch, ensuring proper logic levels are maintained. *M5 and M6:* These are the *access transistors* that connect the internal latch to the bit lines (BL and BLB) during read and write operations. They are controlled by the Word Line (WL).

States of the SRAM Cell

The operation of the SRAM cell can be described in two states:

1. **State 0**
   - Voltage at **X** is *0¨*, and voltage at **Y** is *1¨*.
   - *M5 and M6* are *ON* since the Word Line (WL) is *ON*.
   - *M1 and M4* are *ON* because $\mathbf{X = 0}$ and $\mathbf{Y = 1}$.
   - *M2 and M3* are *OFF* in this state.

2. **State 1s**
   - Voltage at **X** is *1¨*, and voltage at **Y** is *0¨*.
   - *M5 and M6* remain *ON* as WL is *ON*.

- *M2 and M3* are *ON* because $\mathbf{X = 1}$ and $\mathbf{Y = 0}$.
- *M1 and M4* are *OFF* in this state.

State Table

State	X	Y	M1	M2	M3	M4	M5	M6	BL	BLB
0	0	1	1	0	0	1	1	1	0	1
1	1	0	0	1	1	0	1	1	1	0

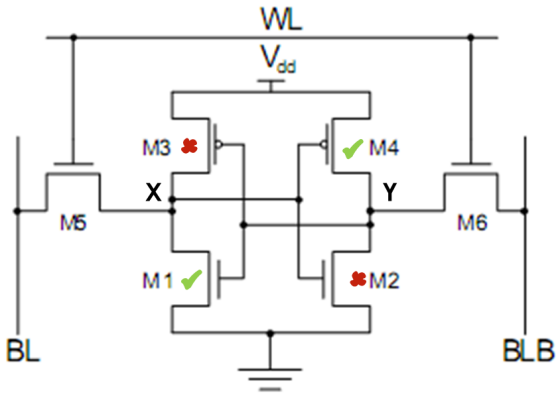

Figure 17: Reading logic 1 in sRAM memory.

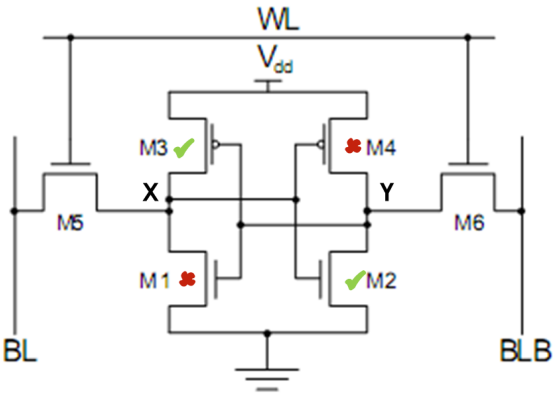

Figure 18: Reading logic 0 in sRAM memory.

5.2 Dynamic RAM

Kindly refer to Fig. 19 for the text below.

Circuit Description

A Dynamic RAM (DRAM) cell consists of a single MOSFET and a capacitor. The MOSFET functions as an access transistor, controlled by the Word Line (WL). The capacitor serves as the storage element, holding the charge that represents binary data, either ”0” or ‘1”. Unlike SRAM, the DRAM cell relies on the temporary nature of charge storage, which makes it compact but necessitates periodic refreshing to maintain the stored information.

Operation of DRAM Cell

The operation of a DRAM cell can be understood through its two primary functions: write and read operations.

During the **write operation**, the Bit Line is driven to either a high voltage (V_{DD}) or ground (GND) to represent data ‘1” or ”0,” respectively. The Word Line is then activated to turn the MOSFET **ON**, allowing the capacitor to charge or discharge accordingly. If the Bit Line is at V_{DD}, the capacitor is charged to store ”1.” If the Bit Line is at GND, the capacitor discharges to store ”0.”

In the **read operation**, the Bit Line is precharged to half the supply voltage ($V_{DD}/2$). When the Word Line is activated, the MOSFET turns **ON**, and the capacitor shares its charge with the Bit Line. If the capacitor was charged (storing ‘1”), the Bit Line voltage increases slightly. If the capacitor was discharged (storing ‘0”), the Bit Line voltage decreases slightly. A sense amplifier detects these small changes in voltage to determine whether the stored value is ‘0” or ”1.” Since the read operation is destructive, the data must be rewritten to the cell after being read.

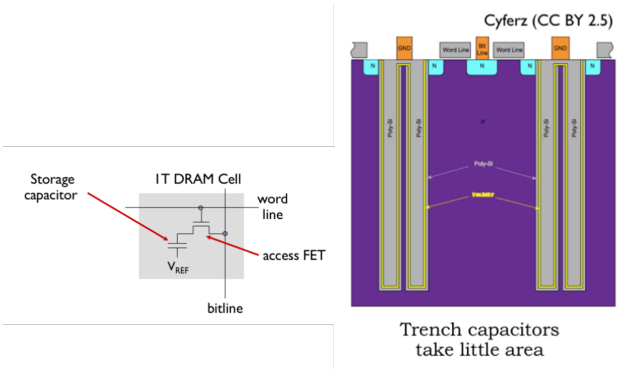

Figure 19: The basic unit cell of DRAM, consisting of a capacitor and a MOSFET, is presented.

Refresh Operation

The charge stored in the DRAM capacitor leaks over time, causing the data to decay. To prevent data loss, periodic refresh operations are necessary. During a refresh, the Word Line is activated to read the existing data, which is then rewritten back into the capacitor. This ensures that the stored charge is restored, maintaining data integrity. Refresh operations are performed automatically by the DRAM controller at regular intervals.

Advantages and Disadvantages

Dynamic RAM offers several advantages over other types of memory, particularly SRAM. The smaller cell size of DRAM allows for higher memory density and lower cost per bit. This makes DRAM an ideal choice for applications requiring large amounts of memory, such as computer main memory.

However, DRAM has some disadvantages. It has slower access times compared to SRAM and requires periodic refreshing, which increases power consumption and system complexity. Despite these drawbacks, its high density and cost-effectiveness make DRAM a widely used memory technology.

5.3 Memory Shadowing

Memory shadowing is a technique used in computer systems to improve the speed of memory access by copying data from slower, non-volatile memory (such as ROM or Flash) into faster, volatile memory (such as RAM). Once the data is copied (or "shadowed") into RAM, the system operates by accessing the data from the RAM instead of the slower memory.

5.4 Why Shadowing is Used?

1. *Speed Advantage:* RAM access is significantly faster than ROM/Flash access.
2. *Execution Efficiency:* Running code from RAM reduces instruction fetch time and accelerates execution.
3. *Real-Time Performance:* Makes firmware more responsive in real-time embedded applications.

The system boots from non-volatile memory and then copies the code into RAM before executing.

1. On reset, the system reads start-up code from non-volatile memory (ROM/Flash).
2. The start-up code copies the ROM contents into a designated RAM region.
3. Once copied, the processor is reconfigured to execute code from this RAM region as though it were the original memory space.

This process may be supported via hardware address decode logic or implemented purely in software.

5.5 Benefits:

- *Improved Execution Time:* Faster fetch and decode cycles from RAM.
- *Resource Overhead:* Requires additional RAM to store the shadowed code.
- *Design Complexity:* Must handle RAM allocation and ensure consistency between original and shadowed memory.

Shadowing trades off memory resource usage for execution speed — a beneficial trade in many embedded designs.

5.6 Memory selection

In embedded system design, choosing the right memory technologies and configurations is important for ensuring optimal performance, cost, and power efficiency. The memory type and size significantly influence system behaviour, real-time capabilities, and application requirements.

Volatile Memory: Loses data when power is removed (e.g., SRAM, DRAM).

Non-volatile Memory: Retains data without power (e.g., Flash, EEPROM, ROM).

Program memory resides in non-volatile memory (Flash/ROM) to store firmware, while *data memory* is held in volatile RAM during execution. Non-volatile RAM types like EEPROM or NVRAM are used for configuration data that must persist across power cycles.

5.7 Key Factors for Memory Selection

When selecting memory components for an embedded system, we have to consider the following criteria:

1. *Speed:* Fast memory improves performance, particularly critical for real-time tasks.
2. *Capacity:* Size of the memory needed based on program code, variables, buffers, and tables.
3. *Interface Type:* Parallel (faster but more pins) vs serial (slower, fewer pins) interfaces.
4. *Power Consumption:* Critical for battery-powered or low-power systems.
5. *Cost:* Higher speed and larger capacity often increase system BOM cost.
6. *Bus Width and Latency:* Wider buses and lower latency support higher throughput.
7. *Non-Volatility Requirements:* Decide between volatile and non-volatile based on data permanence needs.

5.8 Typical Memory Requirements

Embedded systems generally require:

- *Program Memory* – Stores firmware or OS code.
- *Data Memory* – Stores runtime variables and stack.
- *Persistent Storage* – Stores configuration data or look-up tables that must persist across resets.

Two key parameters - memory size and word size. The typical description of memory size will be like 512 kB, 1024 KB, etc are memory chip size (memory density expressed in typical memory bytes per chip. The word size is the number of bits a word is represented in, like 8-bit, 16-bit, 64-bit etc.

6 Memory According to the Type of Interface

The memories are also classified based on the *interface used to communicate* with the processor or controller. The choice of interface impacts system speed, pin count, power consumption, scalability, and overall cost.

Based on how data and addresses are transferred, memory devices are broadly classified into the following categories.

6.1 Parallel Interface Memory

Parallel interface memory transfers multiple bits of data simultaneously using separate data and address lines. Parallel memories use a wide data bus (e.g., 8-bit, 16-bit, or 32-bit) and dedicated address lines for memory access. *Examples:* SRAM, DRAM, Parallel NOR Flash.

Characteristics:

1. High data transfer rate due to simultaneous bit transfer.
2. Requires a large number of pins for data, address, and control signals.
3. Simple access protocol and low latency.

These are used in high-performance embedded systems, microprocessors, and systems requiring fast instruction execution.

6.2 Serial Interface Memory

Serial interface memory transfers data one bit (or a few bits) at a time using a clocked serial protocol. *Examples:* Serial EEPROM, Serial Flash, SD cards.

Serial memories communicate with the processor using fewer pins by sequentially sending data. *Common Serial Interfaces:*

1. SPI (Serial Peripheral Interface)
2. I^2C (Inter-Integrated Circuit)
3. UART-based EEPROMs

Characteristics:

1. Reduced pin count.
2. Lower power consumption.
3. Moderate to low data transfer speed compared to parallel memory.

6.3 Memory-Mapped Interface

In memory-mapped systems, external memories or peripherals share the same address space as internal memory.

Memory devices are accessed using normal load and store instructions, similar to RAM access.

Characteristics:

1. Simplifies programming model.
2. Faster access compared to I/O-mapped techniques.
3. Requires address decoding logic.

They are used in microcontroller-based systems where external RAM or Flash is mapped directly into the processor's address space.

6.4 Bus-Based Interface Memory

Some memories are connected via standard system buses rather than dedicated memory interfaces. Memory devices communicate using standardized bus protocols. *Examples of Bus Interfaces:*

1. PCI / PCIe
2. AXI / AHB (ARM-based systems)
3. USB Mass Storage

Characteristics:

1. High scalability.
2. Protocol overhead compared to direct memory access.
3. Supports multiple devices on the same bus.

Advanced embedded systems, SoCs, and multimedia or networking devices.

7 Processing of Microprocessor

We will see how a microprocessor reads and fetches instructions and data.

7.1 Instruction Execution

The microprocessor's execution of instructions as consisting of several basic stages.

1. *Fetch instruction:* the task of reading the next instruction from memory into the instruction register.
2. *Decode instruction:* the task of determining what operation the instruction in the instruction register represents (e.g., add, move, etc.).
3. *Fetch operands:* the task of moving the instruction's operand data into appropriate registers.
4. *Execute operation:* the task of feeding the appropriate registers through the ALU and back into an appropriate register.
5. *Store results:* the task of writing a register into memory. If each stage takes one clock cycle, then we can see that a single instruction may take several cycles to complete.

7.1.1 Pipelining

Pipelining is a common way to increase the instruction throughput of a microprocessor. Let us assume two people, one washing and the other drying, 8 dishes.

1. In one approach, the first person washes all 8 dishes, and then the second person dries all 8 dishes.
2. The other approach, the second person dries the first dish immediately after it has been washed by the first.

Try to imagine the above for two modes of pipelining. It is really simple. Try to correlate with the modes in Fig. 20.

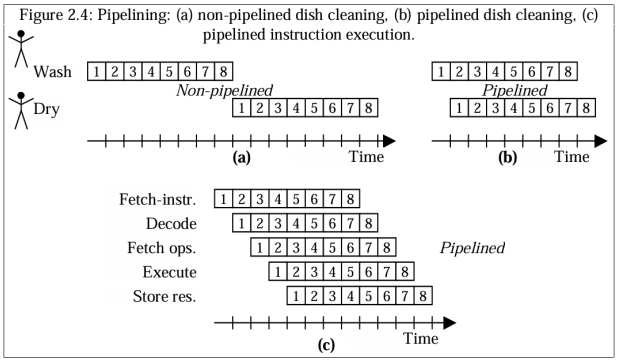

Figure 20: Pipelining (copied figure)

7.2 Instruction Set and addressing modes

An instruction has two parts, an *opcode* field and *operand* fields. An *opcode* specifies the operation to take place during the instruction.The instructions are classified into three categories. *Data-transfer instructions* that move data between memory and registers, between input/output channels and registers, and between registers themselves. *Arithmetic/logical instructions* that configure the ALU to carry out a particular function, channel data from the registers through the ALU, and channel data from the ALU back to a particular register. *Branch instructions* that determine the address of the next program instruction, based possibly on datapath status signals.

Branches can be further categorized as being unconditional jumps, conditional jumps or procedure call and return instructions.

An operand field specifies the location of the actual data that takes part in an operation. The operand field may indicate the data's location through one of several addressing modes, illustrated in Fig. 21.

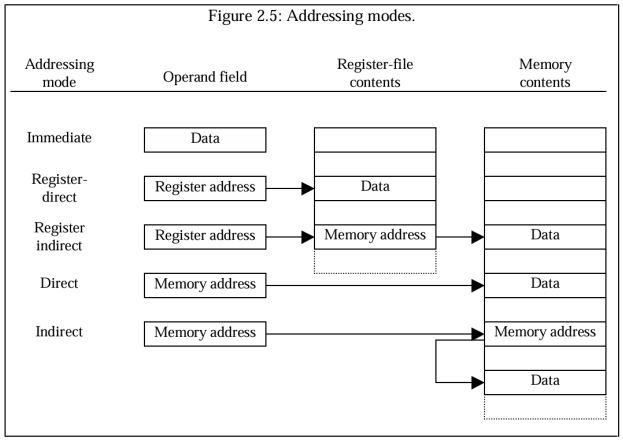

Figure 21: Addressing modes.

8 Other Components:

8.1 Interrupts

An interrupt is a signal to the processor that an important event has occurred. If you type something or a key is pressed, a signal (interrupt) is given to the processor. For every interrupt signal, there is an interrupt service routine (ISR) that will be executed. The processor halts its current operation before implementing

ISR. It pushes its register contents and stack pointer into the stack. There will be a priority list of interrupts to the processor. The assembly-language programmer places each ISR at a specific address in program memory.

8.2 Clock circuitry

The clock circuit consists of a crystal and oscillator and has to be given to the processor. A real-time clock keeps track of dates and time also.

A timer is a device that generates a signal pulse at specified time intervals. The number of clock cycles can be computed using:

$$Numberofclockcycles = Desiredreal - timevalue/Clockcycle \tag{1}$$

A counter is nearly identical to a timer, except that instead of counting clock cycles (pulses on the clock signal), a counter counts pulses on some other input signal. a regular timer. We configure a watchdog timer with a real-time value, just as with a regular timer.

Watchdog timer or reset circuit

The embedded system is reset by a watchdog timer after the timer reaches zero. It is initially set to a high value. A watchdog timer can be thought of as having the inverse functionality than that of a regular timer.

8.3 UART

A UART (Universal Asynchronous Receiver/Transmitter) receives serial data and stores it as parallel data (usually one byte), and takes parallel data and transmits it as serial data. The Baud rate is the transmission and reception rate. indicating the frequency that the signal changes. UART allows full duplex data exchange with external equipment.

8.4 I/O devices

They are classified as Program I/O or Interrupt-driven I/O. The processor sends the data on its own in programmed I/O. In the Interrupt-driven I/o, ISR is executed and the processor is halted. Dynamic Memory Access (DMA) facilitates the transfer of data between the input/ output device and the memory directly without the intervention of the processor. The references followed are [1] [2] [3] [4]

References

[1] D. Hodgson, "Chapter 20 - mechatronics and physical computing," in *Exploring Engineering (Fifth Edition)*, fifth edition ed., P. Kosky, R. Balmer, W. Keat, and G. Wise, Eds. Academic Press, 2021, pp. 453–477. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B978012815073300020X

[2] K. Prasad, *Embedded / Real-Time Systems:*. Wiley, 2003).

[3] K. Shibu, *Introduction to embedded systems*. Tata McGraw-Hill Education, 2009.

[4] S. Chattopadhyay, *Embedded System Design*. PHI Learning Pvt. Ltd., 2023.