# Microprocessor Crash Course!!!

**= must know!

*= very important!

****Exceptions** - On arm processors, interrupts are called exceptions. It is caused when an external hardware activates a specific input line at this forces the microprocessor to interrupt current program to execute special handler routine.

- event that happens. Exception handler runs the code when exception occurs from software errors when undefined instruction is given to processor or invalid memory access.

***CISC Architecture** - Complex Instruction Set Computing where single instructions can execute several low level operations.

***RISC Architecture** - Reduced Instruction Set Computing where CPU design strategy based on the insight that simplified instructions can provide higher performance if it enables faster execution for each instruction.

***CISC and RISC Comparison**

**CISC**- more instructions which increases the complexity of the microcode and slowing down the speed (Intel).

**RISC** - less instructions, less complex code, faster execution ( ARM).

***Superscalar Architecture** - a system that executes multiple instruction at the same time. It fetch and decodes multiple instruction at once.

****IRQ and FIQ Comparison**

**IRQ** - an interrupt request, signal is sent to the processor so the interrupt needs to be handled.

**FIQ** - higher priority interrupt request that is prioritised by disabling IRQ and other FIQ handlers during request servicing. No other interrupts can occur during the processing of active FIQ interrupt.

***Polling** - When a program samples the I/O if an external device. It can be quite wasteful due to having to constantly check for a change in the input.

****Interrupts** - Signals are passed to the processor from an external source. The currently executed code is halted temporarily while code for the interrupt is executed.

***Polling And Interrupt Comparison**

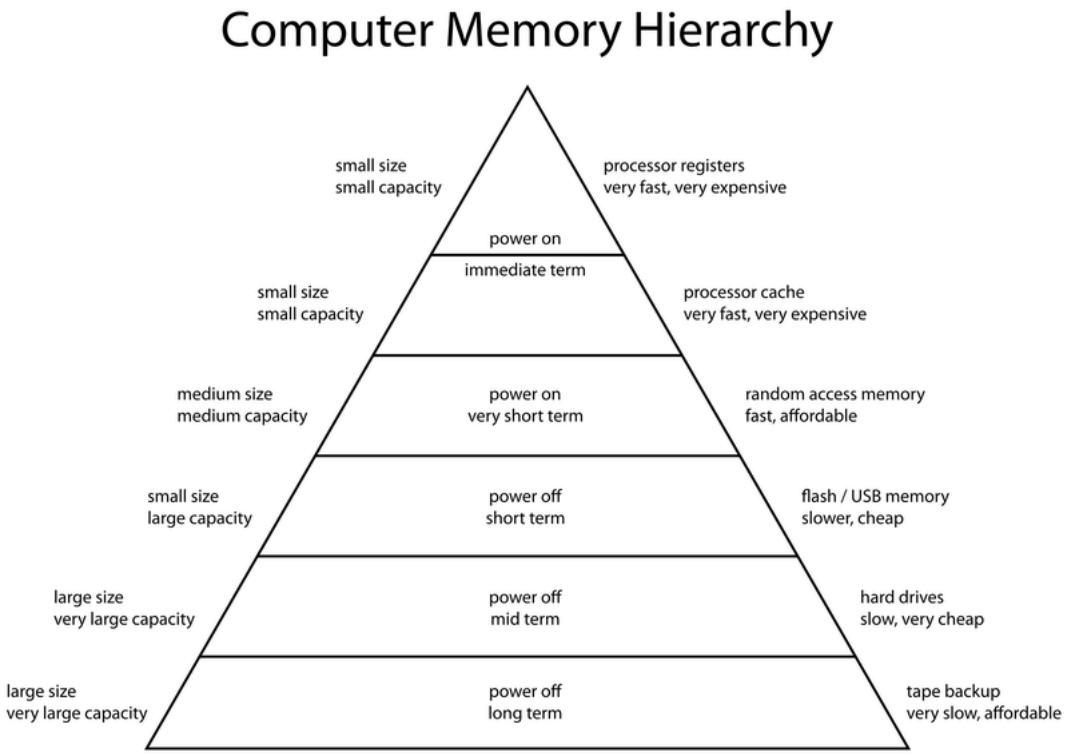
Interrupts is more efficient than polling due to not having to constantly check the state of the external device.

***Vector Interrupt Handling** - an index into an array called an interrupt vector table that contains the memory addresses of interrupt handlers.

***Non Vectored Interrupt** - an interrupt is received from an external device. To find which device caused the interrupt, it is required to look up the interrupt handler that matches the interrupt signal received.

**<u>*Vector Interrupt Handling And Non Vectored Input Comparison</u>**
Vector input handling is better because it is quicker to find the appropriate interrupt handler for the interrupt request.

****CAME UP THREE YEARS IN A ROW!!!****
**** Components And Properties Of Memory Hierarchy Explaining Why It Is Necessary. Describe Function And Operation Of Cache Memories Including The Types of Cash Organisation And Replacement Policies With Advantages And Disadvantages.**




A memory hierarchy in computer storage distinguishes each level in the “hierarchy” by response time. The further away from the processor, the larger the capacity gets. A memory hierarchy is **necessary** to provide storage for large amounts of data and fast processing. To design high performance, the memory hierarchy structure would be compulsory.

The **function** of cache is to provide information in the main memory that is closer to the processor. Cache memory is a high speed memory in the CPU that is used for faster access to the data.

The **operations** include the cache storing blocks of data in block frames. Each block frame has an associated Tag RAM. The tag Ram stores information about which block of memory is stored in the block frame. When the processor looks for data in the memory the cache will now use the tag RAM to check if any block frames contain that address from memory. If data is in the cache then a ‘cache hit’ occurs and the cache provides data. If data was not present a ‘cache miss’ occurs , that data may be stored in the cache for future reference.

**Fully Associative** - Any block can be placed in any block frame. It does simplify adding data to the cache, however when checking if the data from memory is in the cache every block frame must be examined reducing the speed.
**Replacement Policy** : The oldest cache line is evicted from the cache.The policy is called least recently used (LRU).
**Advantage** - solves the potential problem of thrashing with a direct- mapped cache.
**Disadvantage** - the larger the cache, the higher the cost. Additional logic to track usage of lines.

**Direct Mapped** -The memory address of the incoming cache line controls which cache location is going to be used.
**Replacement Policy** - is built in as the cache line replacement is controlled by the memory address.
**Advantage** - simple and efficient organisation, faster than other cache.
**Disadvantage** -potentially replace a cache line that still contains information needed shortly afterwards.

***Polling Template***

```
     AREA MainCode, CODE, READONLY
     IMPORT    main
     EXPORT    start

start
                MOV R0, #0
               ;call the subroutine
                BL subroutine

stop B    stop

     AREA SubroutineCode, CODE, READONLY
subroutine
                ; Return Values
                ; R0 => ASCII code of key pressed
                ; Description
                ;
                 ; save the other registers onto the stack
                STMFD     sp!, {R1-R12, lr}
                ;; polling subroutine code goes here
                 ; restore the registers and the pc
                LDMFD sp!, {R1-R7, pc}
                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                ; END OF SUBROUTINE

     AREA Stuff, DATA, READWRITE

     ;memory to keep track of key presses
BUTTON_ADDRESS DCD 0xE0F05204

END
```

## *Interrupt Template*

```
     AREA InitialisationAndMain, CODE, READONLY
     IMPORT    main

     EXPORT    start
start
; initialisation code
; Initialise the VIC
     ldr  r0,=VIC            ; looking at you, VIC!

     ldr  r1,=irqhan
     str  r1,[r0,#VectAddr0]  ; associate our interrupt handler with Vectored
Interrupt 0

     ;Initialise interrupt handler to be called by the timer (every 16.12356ms)
     ;; Initialise the Timer
     ;; initialise the LED
     ; Initialisation code is finished so loop forever
foreverLoop b foreverLoop

; Interrupt handler code below
AREA InterruptHandler, CODE, READONLY
irqhan    sub  lr,lr,#4
     stmfd     sp!,{r0-r8,lr} ; the lr will be restored to the pc

;this is the body of the interrupt handler
     ;;;;;;;;;;;;;;;;;;;;;;;
     ;; WRITE CODE HERE
     ;;;;;;;;;;;;;;;;;;;;;;;

;this is where we stop the timer from making the interrupt request to the VIC

;here we stop the VIC from making the interrupt request to the CPU:
     ldr  r0,=VIC
     mov  r1,#0
     str  r1,[r0,#VectAddr]   ; reset VIC

     ldmfd     sp!,{r0-r8,pc}^     ; return from interrupt, restoring pc from lr
                    ; and also restoring the CPSR

     AREA Subroutines, CODE, READONLY

     AREA Stuff, DATA, READWRITE

     ;memory to keep track of leds
LED_ADDRESS DCD 0x0E004003A
LED_STATE DCD 0 ; 0 for on 1 for off and 2 for remaining dim
TIMER_COUNT DCD 0;
LED_TOGGLE_COUNT DCD 0;
```

## If You Have Time Left Over!!!

**Three Techniques For Speeding Up Instruction Execution In The Processor With Advantages And Disadvantages.**

**Pipelining**- breaks the operation down into sections so few/no parts of the processor are idle.
**Advantages** - the cycle time of processor is reduced.
- It increases performance.
**Disadvantages** - More transitions per second, move current, higher power.
- Power dissipation of semiconductor devices rises with clock rates.

**Hyper Threading**- duplicates the sections of the processor that store the architectural state.
**Advantages** - reduces the time a processor is idle, appear to be 2 processors and there will be a gain in instructions executed.
**Disadvantages**- can cause cache misses and data dependency.

**Cache**- store data from memory closer to the processor to remove long delays when fetching data from main memory.
**Advantage** - enhance speed of execution, saves time if instructions are repeated.
**Disadvantage** - limited size, difficult to implement, can cause trouble with threading.

**The Difference Between Static And Dynamic Ram. How is Dynamic Ram Organised ?**
**Static RAM**- volatile, used for cache, doesn't have to refresh memory, more hardware and faster than dynamic.
**Dynamic RAM** - volatile, has to refresh memory.
It is **organised** in 2 dimensional matrices with each row being written to or read from at a time. Each cell in the row is indepedent of each other.

**Explain The Concept Of Privilege. What Is The Motivation For It In A Modern Processor? Examples Of Implementation In Arm.**
**Privilege** is a way of preventing programs/users from performing certain actions.
It is about trust and protection.
**Trust** - when a task is to be performed, data is accessed depending on the privilege level. The highest level could access data to any segment needed. At a low privilege level, it can only access segments at the same level or lower privilege level as there is not enough trust.
**Protection** - lowest privilege level - least protected.
highest privilege level - highly protected.
motivation - permission to perform an action.
level 0 -> supervision mode.
level 3 -> application programs, user mode.
Most programs in arm would be executed in user mode , privileged actions are only capable when arm is in supervisor mode.

**An Exception Handler's Return Address May Have To Be altered Before Use. Why? What Alternations Are Needed?**

Due to implementation of a pipeline, the actual value of the program counter may be ahead of the last instruction executed. So when branching back to normal program execution, the program counter may need to be changed. If data abort caused the exception, the program counter is altered to go to the instruction that caused the exception. If undefined instruction caused the exception, no alteration will be needed.

**Explain How A Pipelined Architecture Can Speed Up The Execution Of Instruction On A Processor. Explain The Issues That Can Slow A Pipeline Down And Solutions.**

Each stage of a pipelined architecture ( fetch,decode, execute) must complete its work within one clock period increasing throughput of the system.

**Issue1 - Power Dissipation** ( refer to above)
**Fix** - keep circuitry complexity to a minimum.

**Issue2 - Pipeline Stage Size**. The instructions are broken down into smaller pieces to increase the stage. Stages that take less time will not be running at full utilisation.
**Fix** - keep stages to a minimum. When slowest stage is increased, overall speed increases.

**Issue3 - Stall**, when pipeline cannot complete its work in a clock cycle it 'stalls' the pipeline. Instructions close to the end will continue, instructions in stages behind the stalled stage are stopped.
**Fix** - branch prediction attempts to reduce the number of stalls by predicting whether a branch will be taken or not.

**The Four Stages Of A Pipeline.**

**Fetch**- instruction from address stored in the counter, that instruction loads in to register.
**Decode**-Instruction in register.
**Encode**- Execute instructresion.
**Write**- Write to register.

**Explain The Approaches Of Polling And Interrupt Handling To Detect Activity On Peripheral Interfaces And The Difference Between Them.**

( refer to above )
polling - expensive , delay depends on how often the polling is done.
interrupt - complex to implement , fastest response to even.

**What Is Meant By Latency In The Context Of An Interrupt Handler?**

In the contact of an interrupt handler latency is the extra time delay introduced by getting to the interrupt handler itself.

**Explain The Terms Stack Based Architecture And Register Based Architecture. What Are The Relative Merits Of Each?**

**Stack Based Register** - memory structure where operated are stored in a stack data structure. Operations are carried out by popping data from stack and pushing the results in LIFO.
**Advantage** - simple instruction set, operands are addressed by stack pointer.
**Disadvantage** - good compilers can do well with registers.

**Register Based Architecture** - the data structure where the operands are stored is based on the registers of the CPU.No push or pop operations
**Advantage** - allows optimizations that can not be done in stack based approach, overhead of pushing to and popping from is non existent.
**Disadvantage** - the average register instruction is larger than an average stack instruction.

**Returning From An Exception Is Tricky. What Are The Difficulties And Solutions?**
**-** restore the CPSR from spsr_mode ,restoring program counter using return address stored in lr_mode.
**-** Adjust exception return address, depending on type of exception, return address would be subtracted by appropriate value.

**What Is An Instruction Set ( ISA)? What Is The Relationship Between An ISA And A Processor's Instruction Set?**
ISA is part of the computer architecture that is related to programming, including data types, instructions , registers, memory architecture, addressing modes, interrupt, exception handling and external I/O.
Processor's Instruction Set - the set of processor design techniques used to implement instruction set.

**Define Superscalar Architecture. How Does It Relate To The Sequential Execution Semantics Of A Conventional Processor? What Problems Are There With Superscalar Architecture?**
Definition - refer to above.
A superscalar CPU architecture implements instruction level parallelism within a single processor. It allows faster CPU throughput.

Problem - if one of the instructions is branched then we will be moved to a different part of memory and all the other instructions would be dropped.
- wastes fetch bandwidth.