Arm ISA

- Arm has 2 instruction sets (in general for Cortex M) 1) ARMv7 - Standard ISA (32bit)  
2)Thumb - incorporates 16bit/32bit mix using a **unified assembler language (UAL)** to alternate btw the 2 ISAs. It adds a suffix to the end of the instruction 'N'arrow -16bit; 'W'ide -32bit. Each Thumb instruction directly correlates to a 32bit equivalent ARM instruction, just their encoding is different.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | OP | IMM | Rs | Rd |

2b 5b 3b 3b

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Cond | 0 | 0 | 1 | OP | S | Rn | Rd | Rotate | Imm |

4b 4b 4b 4b 4b 8b

Why 16b? easier to decode?

I$ (inst cache) =32b=> IF -> iD less power by fetching double instructions, using the memory twice. sacrifice precision.

|  |  |
| --- | --- |
| R0 | Low Registers |
| R1 |
| R2 |
| R3 |
| R4 |
| R5 |
| R6 |
| R7 |
| R8 | High Registers |
| R8 |
| R9 |
| R10 |
| R11 |
| R12 |
| R13: Stack pointer | Used by OS kernel and Stack |
| R14: Link Register | Used when subroutine is called |
| R15: Program Counter | Implements PC in Regi file |

Thumb

ARM v7

E  
  
PROGRAM STATUS REGISTER (xPSR) : Keeps track of arithmetic and logic processing flags, exe & interrupts Conditional Exec

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ARM** | N | Z | C | V | Q | IT | J | (Res) | GE[3:0] | IT | A | F | T |
| **Thumb** | " | " | " | " |  |  |  |  |  |  |  | " | " |
|  | Negative | Zero | Carry | Overflow | Satu |  | Jazell |  | >= |  |  | FIQ | thumb |

Other Registers  
PRIMASK  
FAULTMASK interrupt handling  
BASEPRI  
CONTROL - status tracker and SP

Program counter is in the Register file - save power.

EXAMPLE:

|  |  |
| --- | --- |
| Subroutine\_ex() PC+4 : //code void Subroutine\_ex(){  int a = b + 1; } | MOV R14 R15 **PC to LINKREG (keep spot)**  Push {R1}; **R13 = R13 - 4 (R13 is SP) || M[SP] = R1**  ADD R3 R1 #1;  POP {R1}; **R13 - R13 + 4 || R1 = M[SP]**  BX R14 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | <-SP |  |  |  |
| Reg Contents | -4 | Data Processing | POP | Reg Contents |
| <-SP |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

1) Call function  
2) Move PC to LR, Call Subroutine  
3) Push Registers to Stack  
4) Pop Stack  
5) Branch Back to address saved in LR

Cortex-M0 - lowest power

Rev Instruction

Word is 32-bit