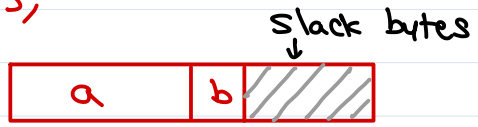
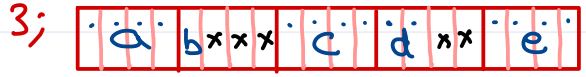
Advanced Micro-controllers - ARM

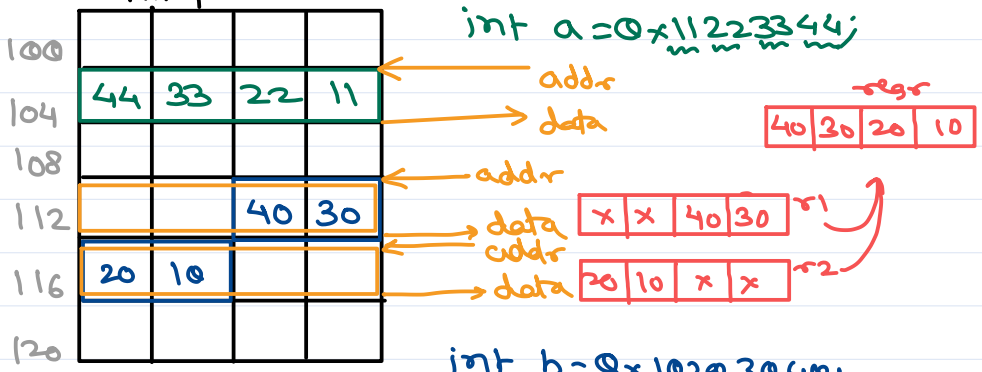
DESD @ Sunbeam Infotech

```
struct test {
    int a;
    char b;
};
```




```
struct test2 {
    int a;
    char b;
    int c;
    short d;
    int e;
};
```






```
int a = 0x11223344;
```

addr

data

addr

data

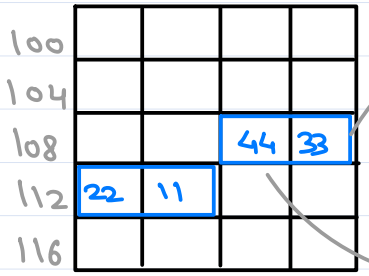
addr

data

reg


```
int b = 0x10203040;
```

ARMv4




```
int a = 0x11223344;
&a = 110.
pf("%x", a);
    → 11223344

char c1 = 0x44, c2 = 0x33,
     c3 = 0x22, c4 = 0x11;
int x = c1 | (c2<<8) |
        (c3<<16) | (c4<<24);
```

ARM Ltd

Acorn Computers (UK)
↳ New processor → RISC → 32-bit CPU
↳ ARM
↳ Acorn RISC Machine

- ARM founded in November 1990
  - **A**dvanced **R**ISC **M**achines
- Company headquarters in Cambridge, UK
  - Processor design centers in Cambridge, Austin, and Sophia Antipolis
  - Sales, support, and engineering offices all over the world
- Best known for its range of RISC processor cores designs
  - Other products – fabric IP, software tools, models, cell libraries - to help partners develop and ship ARM-based SoCs
- ARM does *not* manufacture silicon
- More information about ARM and our offices on our web site:
  - http://www.arm.com/aboutarm/

ARM Offices Worldwide

ARM Connected Community – 900+

Connect, Collaborate, Create – accelerating innovation

ISA vs Core

ISA = Instruction Set Architecture.

- ISA → Instruction Execution Engine
  - ↳ opcode + operand
- Executes machine level instructions.
- Versions: v1, v2, v3, v4, v5, v6, v7, v8.
- a.k.a. micro-architecture.
- Doesn't include Flash, RAM, Cache, MPU/MMU.

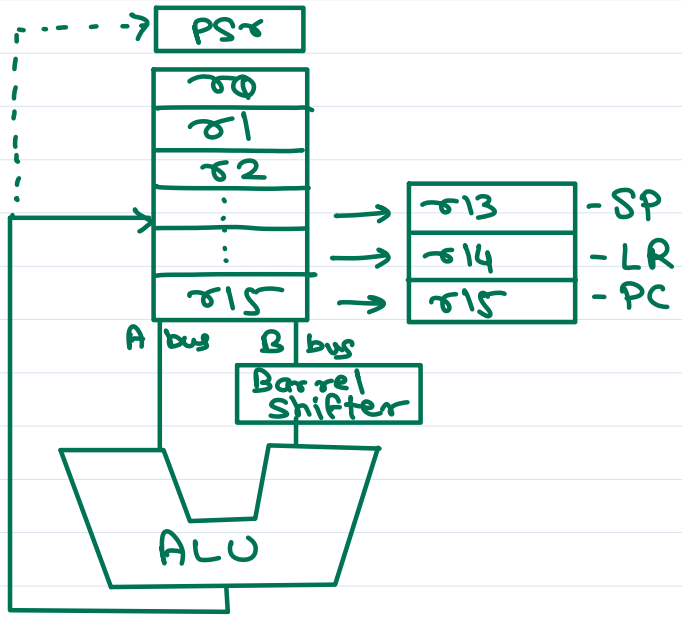
Core = ISA + Cache + MPU/MMU + Debug facility + buses + ...

- based on some ISA.
- versions: ARM7, ARM9, ARM10, ARM11, Cortex-A8, Cortex-M3, Cortex-M4, ...
- ARM v4 → ARM7 (~~MPU~~) and ARM9 (MMU)

MC = Core + Flash + SRAM + Peripherals.

- e.g. STM32F407VG (CM4)
  - LPC1768 (CM3)
  - LPC2148 (ARM7)
  - AM335x (CA8)

In general, ARM core (basic).



ARM arch

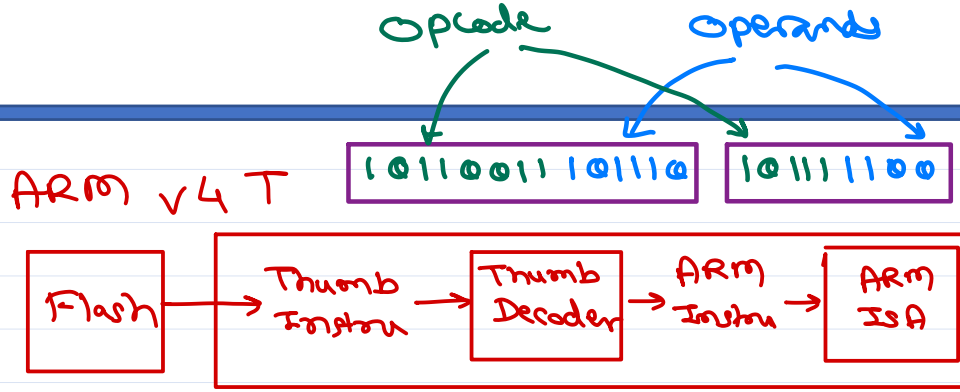
word = 32 bit

half-word = 16 bit

byte = 8 bit

ARM Instructions

① ARM Instruction Set
- each instruction is 32-bit
  ↳ ADDEQ R0,R1,R2 | ADD R0, R1, R2, LSL #2 <<
② Thumb Instruction Set
- each instruction is 16-bit
- increased code density
- Thumb limitations:
  ① cannot used for exception handling
  ② cannot manipulate status register
  ③ not all instrn can access all registers (limited regs: r0-r7)
  ④ Conditional exec not possible.
  ⑤ Inline barrel shift not supported.
- Needs 70% less code space.



SLOS: Thumb encoding | ARM encoding

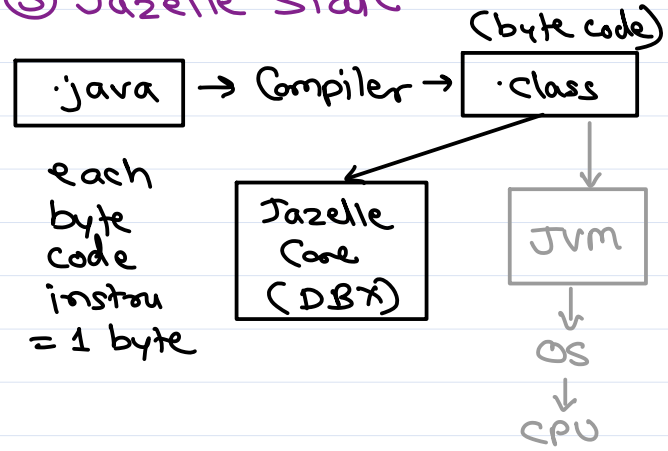
ARM processor state

① ARM state
- fetch 4 byte instrn from flash
- PC = PC + 4
- PC is multiple of 4 [31.... 2 0 0]

② Thumb state
- fetch 2 byte instrn from flash
- PC = PC + 2
- PC is multiple of 2 [31..... 1 0]

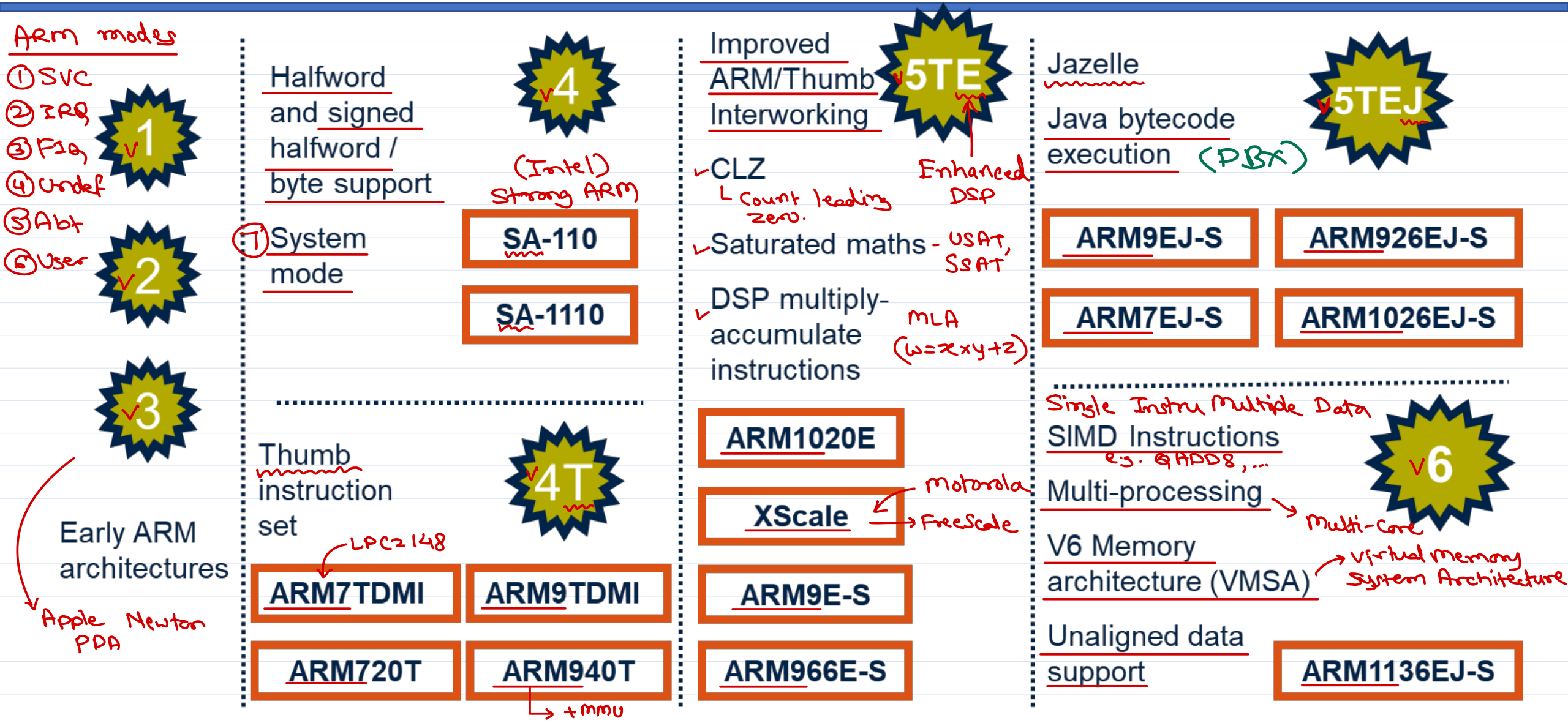
ARM - Thumb interworking : Instructions to switch from ARM to Thumb state & vice versa - BX label, BLX func
label/func → addr → is loaded in PC & last bit (0) indicate state → 0: ARM state, 1: Thumb state

③ Jazelle state



✓ PC is multiple of 4.
✓ each time 4 instrns (4 bytes) are fetched from code memory.

Development of the ARM Architecture



Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>