Single Cycle Implementation of Subset of ARM instructions

Instruction set and its encoding

Data Processing instructions

We consider instructions with two operand and one result, and

1. ADD Rd, Rn, Rm | ADD Rd, Rn, #imm
2. SUB Rd, Rn, Rm | SUB Rd, Rn, #imm
3. AND Rd, Rn, Rm | AND Rd, Rn, #imm
4. ORR Rd, Rn, Rm | ORR Rd, Rn, #imm
5. ADC Rd, Rn, Rm | ORR Rd, Rn, #imm
6. CMP Rn, Rm | CMP Rn, #imm
7. MOV Rd, Rm | MOV Rd, #imm
8. MNV Rd, Rm | MNV RD, #imm

Note that:

- We support only two formats for each operation/opcode.
    - Second operand be a register, or an immediate value
- Shift operands are not supported.
- Immediate value should be less than or equal to 255
- Update flags is not supported except for CMP instruction

Encoding:

Cond	F	I	Opcode	S	Rn	Rd	Operand2
4 bits	2 bits	1 bit	4 bits	1 bit	4 bits	4 bits	12 bits

Figure 1 Encoding for data processing instructions

- Cond (bits 28-31): Bits are set according to figure 3 for conditional execution. For our implementation its value should be 14.
- F (bits 26-27) : Value if 0 for these set of operations
- I (bit 25) : Value 0 if second operand is register, 1 if second operand is immediate
- Opcode (Bits 21 to 24)
    - 0 for AND

    - 1 for EOR (xor operation)
    - 2 for SUB
    - 4 for ADD
    - 5 for ADC (add with carry)
    - 10 for CMP
    - 12 for ORR
    - 13 for MOV
    - 15 for MNV (bitwise not)
- S (bit 20) : Value 1 indicate flags are updated. For this design, S will be 1 for CMP instruction otherwise remain 0.
- Rn (Bit 16 to 19) : register specifying first operand
    - For MOV/MNV instruction its value is 0
- Rd (bit 12 to 15): register specifying destination
    - For CMP instruction, its value is 0
- Operand2 (0 to 11)
    - Bit 0 to 3 : for register operand, rest is ignored for this implementation
    - Bits 0 to 7 : for immediate operand, rest is ignored for this implementation

Data –transfer instructions

For this implantation, we consider two instructions:

1. LDR Rd [Rn, #imm]
2. STR Rd [Rn, #imm]

Note :

- Only immediate offset addressing mode is considered for this implementation. All other addressing modes are ignored.
- Immediate value should be less than or equal to 4095
- Rn is the base address for load/store instruction.
- Only 32 bit load/stores are considered for this implementation, that is half word, byte or double word is not considered.

Encoding:

Cond	F	Opcode	Rn	Rd	Offset12
4 bits	2 bits	6 bits	4 bits	4 bits	12 bits

Figure 2 Encoding for DT instruction

Field values:

- Cond (bits 28-31) : 14 for this implementation
- F (bits 26-27) : 1 for DT
- Opcode (bits 20-25)

    - 25 – for LDR
    - 24 – for STR
- Rn (bits 16 -19) : register specifying base address
- Rd (bits 12- 15)
    - Destination register for load
    - Register to be stored in memory
- Offset12(bits 0 to 11)
    - Maximum value should be 4095.

Branch instruction

- B<cond> offset

Value	Meaning	Value	Meaning
0	EQ (EQual)	8	HI (unsigned HIgher)
1	NE (Not Equal)	9	LS (unsigned Lower or Same)
2	HS (unsigned Higher or Same)	10	GE (signed Greater than or Equal)
3	LO (unsigned LOwer)	11	LT (signed Less Than)
4	MI (MInus, <0)	12	GT (signed Greater Than)
5	PL - (PLus, >=0)	13	LE (signed Less Than or Equal)
6	VS (oVerflow Set, overflow)	14	AL (Always)
7	VC (oVerflow Clear, no overflow)	15	NV (reserved)

Figure 3 Condition encoding

Following conditions are implemented:

- EQ
- NE
- LT
- LE
- GT
- GE
- AL

Thus following branch instructions are valid: BEQ, BNE, BLT, BGT, BLE, BGE, B

Encoding

Cond	F	Opcode	Offset
4bits	2 bits	2 bits	24 bits

Field values:

- Cond (28 to 31 bits)
- F (26-27 bits) : 2 for branch instruction
- Opcode (bits 24 – 25) : 2 for branch

- Offset(bits 0 to 23).

Branch will change the PC to a new address calculated by :

PC = PC + SignExt32(offset x 4) + 8

Offset is first multiplied by 4, than sign extended for 32 bits.

Design of Single cycle processor

Based on the encoding and subset above, the design of single cycle processor can be summarized in the figure below: