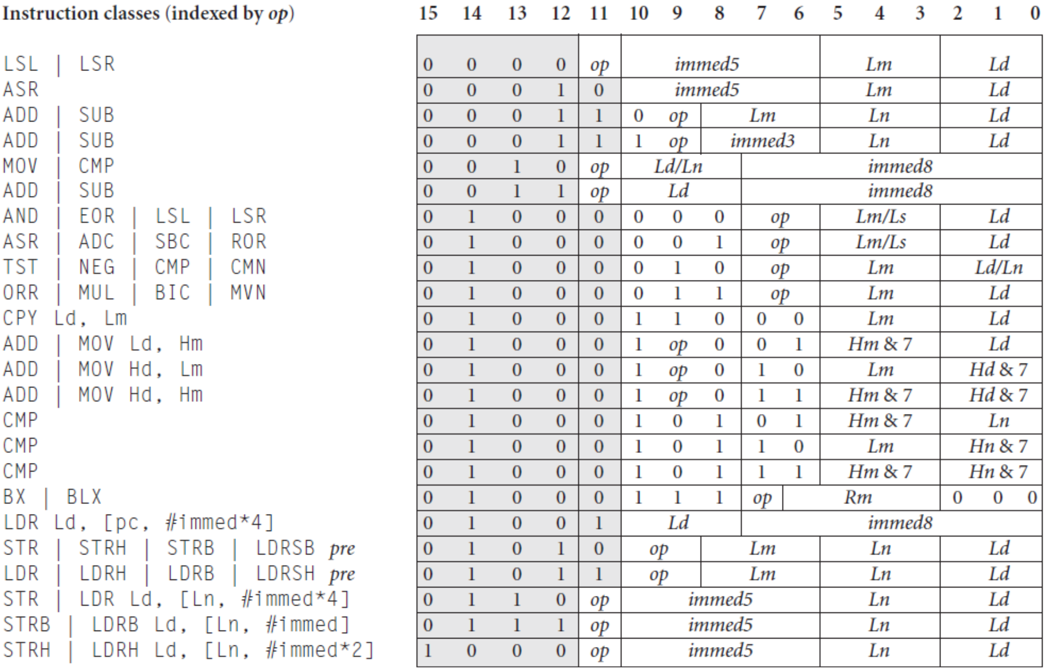
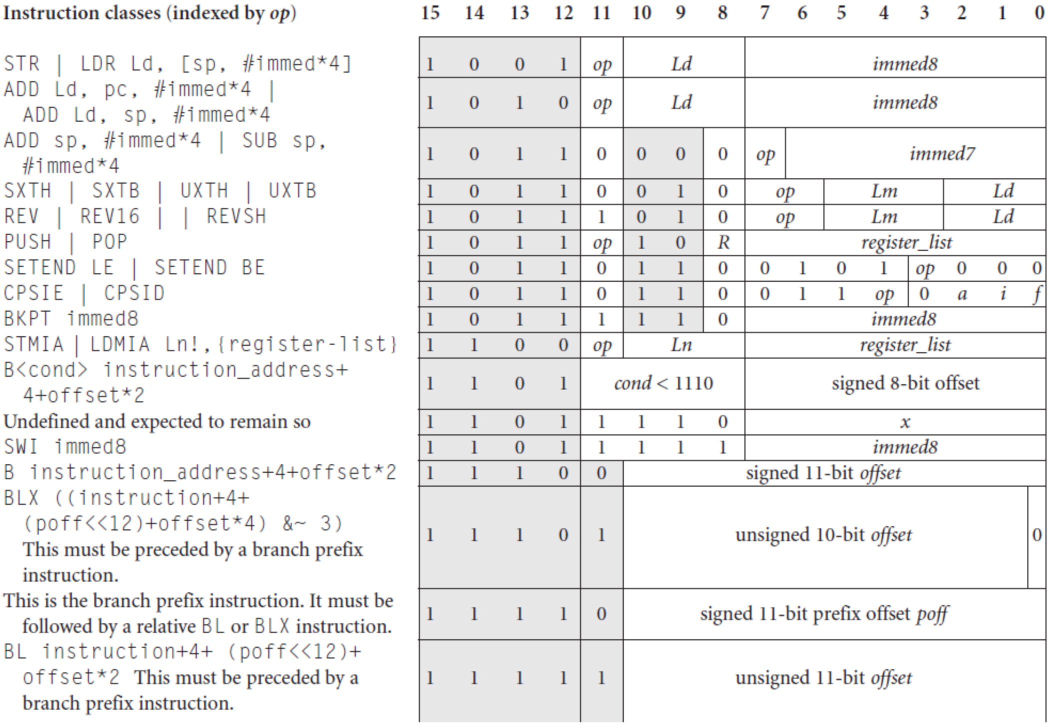
HW3_Pipelined THUMB CPU

Handout: 2024/10/22
Due: 2024/11/05

The course web site has RTL codes of a pipelined microprocessor that can execute 16-bit THUMB instructions. The following lists the THUMB instruction encoding.



Instruction classes (indexed by *op*)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LSL | LSR	0	0	0	0	*op*	*immed5*					*Lm*			*Ld*		
ASR	0	0	0	1	0	*immed5*					*Lm*			*Ld*		
ADD | SUB	0	0	0	1	1	0	*op*	*Lm*			*Ln*			*Ld*		
ADD | SUB	0	0	0	1	1	1	*op*	*immed3*			*Ln*			*Ld*		
MOV | CMP	0	0	1	0	*op*	*Ld/Ln*			*immed8*							
ADD | SUB	0	0	1	1	*op*	*Ld*			*immed8*							
AND | EOR | LSL | LSR	0	1	0	0	0	0	0	0	*op*		*Lm/Ls*			*Ld*		
ASR | ADC | SBC | ROR	0	1	0	0	0	0	0	1	*op*		*Lm/Ls*			*Ld*		
TST | NEG | CMP | CMN	0	1	0	0	0	0	1	0	*op*		*Lm*			*Ld/Ln*		
ORR | MUL | BIC | MVN	0	1	0	0	0	0	1	1	*op*		*Lm*			*Ld*		
CPY Ld, Lm	0	1	0	0	0	1	1	0	0	0	*Lm*			*Ld*		
ADD | MOV Ld, Hm	0	1	0	0	0	1	*op*	0	0	1	*Hm* & 7			*Ld*		
ADD | MOV Hd, Lm	0	1	0	0	0	1	*op*	0	1	0	*Lm*			*Hd* & 7		
ADD | MOV Hd, Hm	0	1	0	0	0	1	*op*	0	1	1	*Hm* & 7			*Hd* & 7		
CMP	0	1	0	0	0	1	0	1	0	1	*Hm* & 7			*Ln*		
CMP	0	1	0	0	0	1	0	1	1	0	*Lm*			*Hn* & 7		
CMP	0	1	0	0	0	1	0	1	1	1	*Hm* & 7			*Hn* & 7		
BX | BLX	0	1	0	0	0	1	1	1	*op*	*Rm*				0	0	0
LDR Ld, [pc, #immed*4]	0	1	0	0	1	*Ld*			*immed8*							
STR | STRH | STRB | LDRSB *pre*	0	1	0	1	0	*op*		*Lm*			*Ln*			*Ld*		
LDR | LDRH | LDRB | LDRSH *pre*	0	1	0	1	1	*op*		*Lm*			*Ln*			*Ld*		
STR | LDR Ld, [Ln, #immed*4]	0	1	1	0	*op*	*immed5*					*Ln*			*Ld*		
STRB | LDRB Ld, [Ln, #immed]	0	1	1	1	*op*	*immed5*					*Ln*			*Ld*		
STRH | LDRH Ld, [Ln, #immed*2]	1	0	0	0	*op*	*immed5*					*Ln*			*Ld*		



Instruction classes (indexed by *op*)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STR | LDR Ld, [sp, #immed*4]	1	0	0	1	*op*	*Ld*			*immed8*							
ADD Ld, pc, #immed*4 | ADD Ld, sp, #immed*4	1	0	1	0	*op*	*Ld*			*immed8*							
ADD sp, #immed*4 | SUB sp, #immed*4	1	0	1	1	0	0	0	0	*op*	*immed7*						
SXTH | SXTB | UXTH | UXTB	1	0	1	1	0	0	1	0	*op*		*Lm*			*Ld*		
REV | REV16 | | REVSH	1	0	1	1	1	0	1	0	*op*		*Lm*			*Ld*		
PUSH | POP	1	0	1	1	*op*	1	0	*R*	*register_list*							
SETEND LE | SETEND BE	1	0	1	1	0	1	1	0	0	1	0	1	*op*	0	0	0
CPSIE | CPSID	1	0	1	1	0	1	1	0	0	1	1	*op*	0	*a*	*i*	*f*
BKPT immed8	1	0	1	1	1	1	1	0	*immed8*							
STMIA | LDMIA Ln!,{register-list}	1	1	0	0	*op*	*Ln*			*register_list*							
B<cond> instruction_address+4+offset*2	1	1	0	1	*cond* < 1110				signed 8-bit offset							
Undefined and expected to remain so	1	1	0	1	1	1	1	0	*x*							
SWI immed8	1	1	0	1	1	1	1	1	*immed8*							
B instruction_address+4+offset*2	1	1	1	0	0	signed 11-bit *offset*										
BLX ((instruction+4+(poff<<12)+offset*4) &~ 3) This must be preceded by a branch prefix instruction.	1	1	1	0	1	unsigned 10-bit *offset*										0
This is the branch prefix instruction. It must be followed by a relative BL or BLX instruction.	1	1	1	1	0	signed 11-bit prefix offset *poff*										
BL instruction+4+ (poff<<12)+offset*2 This must be preceded by a branch prefix instruction.	1	1	1	1	1	unsigned 11-bit *offset*										

1. Trace the given Verilog RTL codes of the 16-bit pipelined THUMB processor, and use the given testbench tb_thumb.v to verify the RTL code. Make necessary modifications for the

codes to make them functionable in both RTL and gate-level.

2. In the original Verilog, all the four pipelined stages (IF stage, ID stage, EX stage, WB stage) are in a single module. Modify the code so that each pipelined stage is in a separate module, so that the critical path delay of each pipelined stage can be easily measured from the synthesis results of the Synopsys Design Compiler (DC).
3. Generate three different synthesis results using different design constraints: area-optimized result, delay-optimized result, and in-between using different constraints in the Synopsys DC. Compare the differences of critical path delays, area, and power for the three different synthesis results.
4. Measure the critical path delay of each pipelined stage in the synthesis results. That is, for each synthesis result, you should measure the critical path delays for the four pipelined stages. And you should do the measurement for the three different synthesis results (area-optimized, delay-optimized, and in-between).
5. Use PrimeTime (PT) to measure the critical path delay and power by providing a sequence of inputs. Compare the delay and power with those obtained from Synospsy Design Compiler (DC). Fill in the following comparison table.

constraint	area	Delay (DC)					power	
		1st	2nd	3rd	4th	Critical	DC	PT
delay-opt						DC (PT)		
area-opt								
In=betwen								

6. Perform automatic placement-and-routing for the THUMB CPU. Mark the four pipelined stages in the layout view.

References:

1. S. Lee, *Advanced Digital Logic Design Using Verilog, State Machines, and Synthesis for FPGAs*, Nelson, 2006. (Chap. 9: Verilog code of the pipelined 16-bit THUMB CPU; Appendix A: THUMB instructions)

Report Requirement

檔案請依以下格式繳交(100%)

HDL_HW3_MXXXXXXXXX.zip / HDL_HW3_BXXXXXXXXX.zip

-rtl 資料夾

--thumb_pipe.v (修正、切 4 pipeline stages : IF、ID、EX、WB) (10%)

-gate 資料夾(合成前述 thumb_pipe.v) (15%)

--area 資料夾

---thumb_pipe_area.v

---thumb_pipe_area.ddc

--- thumb_pipe_area.sdf

--- thumb_pipe_area.sdc

--delay 資料夾

---thumb_pipe_delay.v

---thumb_pipe_delay.ddc

--- thumb_pipe_delay.sdf

--- thumb_pipe_delay.sdc

--between 資料夾

---thumb_pipe_between.v

---thumb_pipe_between.ddc

--- thumb_pipe_between.sdf

--- thumb_pipe_between.sdc

-APR(35%)

-word 報告

--第一部分:

---rtl level 切 4 階 pipeline 的 testbench 波型圖一張(5%)

---gate level 切 4 階 pipeline 的 testbench 波型圖一張(5%)

---以上波型解釋(5%)

--第二部分:

---synthesis area information, individual critical path delays of each pipelined stage，power in both design compiler and PrimeTime 完成以下表格

---表格中，以 mid(in-between)合成結果之 area(x1)、各階 delay(x4)、power(DC、PT)(x2)，report 報告共 7 張截圖(15%)

--第三部分:

---APR 結果圖(5%)

--第四部分:

---心得(5%)