Homework 3 – Pipelined THUMB CPU

Due Nov. 15, 2016

The course ftp side 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 test bench 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. Explain the function of the testbench in the file tb_thumb.v.

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)