Estados da Control

- Fetch
- Decode:
    1. Tipo D - STUR: STURB, STURH, STURW, STUR (ativar o monitor use instruction[29] e para ir para o STXR_EXECUTE use instruction[29:28] & instruction[23:22]).
    2. Tipo D - LDUR: LDURD, LDURH, LDURSW, LDUR, LDXR e STXR((ativar o monitor use instruction[29]).
    3. Tipo R e I: ADD(I)(S), AND(I)(S), SUB(I)(S) ,ORR(I), EOR(I), LSR, LSL(para diferenciar o alu_b_src use instruction[25], para setar flags instruction[29], mas cuidado com ORRI(prioridade para esse bit, mas use instruction[30] e instruction[24]).
    4. Tipo IW: MOVZ, MOVK(DF diferencia elas, basta habilitar o enable).
    5. Tipo B(exceto BL) e CB (use instruction[31:30] & instruction[25]).
    6. BL
    7. MUL_DIV: MUL, SMULH, UMULH, SDIV, UDIV.
    8. BR
- STXR_Execute (faz a escrita, caso stxr_try do ciclo anterior seja 1).
- Três procedures, um para cada busy.

Notas:

- Em 4 e 5 do Decode, separo BL dos demais, pois BL e CBZ/CBNZ usam a ula principal de maneiras diferentes e o BL também escreve no banco de registradores, então resolvi isolar BL.
- Separei 6 de 2, devido ao procedure.

Dicas:

- Faça data_memory_src combinatório(instruction[31:30]) , mul_div_src combinatório (instruction[24] and instruction[22]).
- Deixe pc + 4 pronto na saída da alu_pc após o fetch.

ALU Control

alu_op	alu_control
00	000
01	011
10	100
11	---

Notas:

- alu_op = 00, abrange os subestados do Decode (1), (2) e o STXR_EXECUTE.
- alu_op = 01, abrange os subestados do Decode (4), (6) e (8).
- alu_op = 10, abrange o subestado do Decode (5)(veja que apenas CBZ/CBNZ de fato usam a ula).
- alu_op = 11, abrange o subestado do Decode (3), para decidir o alu_control, observe o opcode, usando um esquemático semelhante ao esboçado para o sign_extended.
- O subestado (7) do Decode não usa a ula principal.