# Why?

You've built a processor that can execute programs consisting of add register and add immediate. How do we add more instructions, like load word/store word and branches? How do we calculate control signals?

# Model 1: Adding ldr to the datapath

Consider the RTL for add register, add immediate, and ldr from the reference sheet.

|  |  |
| --- | --- |
| **Instruction** | **RTL** |
| add Rd, Rn, Rm | R[Rd] <- R[Rn] + R[Rm] |
| add Rd, Rn, constant | R[Rd] <- R[Rn] + ZeroExt(imm8) |
| ldr Rd, [Rn, constant] | R[Rd] <- Mem[ R[Rn] + ZeroExt(imm12) ] |

Note: we are currently only considering ldr with a second operand being a positive constant.

1. On the RTL, identify (e.g., circle) all the differences **between add register and add immediate**.
2. Explain how those differences were implemented in the add processor.

Multiplexor picks between R[Rm[ and ZeroExt(imm8).

1. On the RTL, identify (e.g., circle) all the differences **between add immediate and ldr**.
2. According to the RTL for ldr...

which register is written? Rd

which register(s) is(are) read? Rn

is memory written or read? read

what memory address is accessed? Rn+ZeroExt(imm12)

Now it is time to modify the processor to support ldr. Questions 5-11 will guide you.

1. On your processor diagram (currently supports add register and add immediate), identify with the label **R[Rn]** the wire that carries **R[Rn].**
2. Identify with the label **imm12**the wire that carries **imm12.**
3. Identify with the label **R[Rn] + ZeroExt(imm12)** the wire that carries **R[rs] + ZeroExt(imm12).**
4. Wire up **R[Rn] + ZeroExt(imm12)** with the appropriate port of the data memory.
5. Identify with the label **Mem[ R[Rn] + ZeroExt(imm12) ]** the wire that carries **Mem[ R[Rn] + ZeroExt(imm12) ].**
6. Which port of the register file must take **Mem[ R[Rn] + ZeroExt(imm12)]** as an input?
7. How will you attach **Mem[ R[Rn] + ZeroExt(imm12) ]** to the register file without breaking the other instructions? Do it.

# Model 2: Adding str to the datapath

Consider the RTL for add, ldr, and str.

|  |  |
| --- | --- |
| **Instruction** | **RTL** |
| add Rd, Rn, Rm | R[Rd] <- R[Rn] + R[Rm] |
| add Rd, Rn, constant | R[Rd] <- R[Rn] + ZeroExt(imm8) |
| ldr Rd, [Rn, constant] | R[Rd] <- Mem[ R[Rn] + ZeroExt(imm12) ] |
| str, imm(Rn) | Mem[ R[Rn] + ZeroExt(imm12) ] <- R[Rd] |

Note: we are currently only considering ldr/str with a second operand being a positive constant.

1. What are the similarities between the RTL for ldr and str?

Both use R[Rn] + ZeroExt(imm12).

1. What are the differences between the RTL for ldr and str?

R[Rd] is on the right side, instead of the left side. (This means we are writing to memory)

1. According to the RTL for str...

which register is written? none

which register(s) is(are) read? Rd

is memory written or read? written

what memory address is accessed? R[Rn] + ZeroExt(imm12)

Now it is time to modify the processor to support str. The questions below will guide you.

1. Wire up **R[Rn] + ZeroExt(imm12)** with the appropriate port of the data memory.
2. Modify the processor so that **R[Rd]** is available.
3. Wire up **R[Rd]** to the appropriate port of the data memory.
4. Add new outputs from Control going to

* any disconnected mux selects
* the write-enable input of the register file
* the write-enable input of the data memory

# Model 3: Control

How do we set the values of those mux selects and write-enables? We'll find out next.

Below is a truth table relating the input (opcode) to the outputs (columns to the right, i.e. control signals).

**Prepare:** On your processor diagram, relabel the control signals as shown in class (or textbook ch 7). They should match the output column names in the truth table.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Inputs | | |  |  | | Outputs | | | | | |
| Instruction | Op | I | L | Branch | RegWrite | ImmSrc | | ALUSrc | MemWrite | RegSrc | MemToReg | ALU control |
| Data-processing; Src2 is register | 002 | 0 | NA | 0 | 1 | XX | | 0 | 0 | 00 | 0 | depends on cmd |
| Data-processing; Src2 is immediate | 002 | 1 | NA | 0 | 1 | 00 | | 1 | 0 | X0 | 0 | depends on cmd |
| ldr | 01 | NA | 1 | 0 | 1 | 01 | | 1 | 0 | X0 | 1 | Add (7) |
| str | 01 | NA | 0 | 0 | 0 | 01 | | 1 | 1 | 10 | X | Add (7) |
| B | 10 | X | X | 1 | 0 | 10 | | 1 | 0 | X1 | 0 |  |

Legend: **NA** means this input is not applicable for that instruction. **X** means this output does not matter for this instruction, i.e., it can be made to be either 0 or 1 because it will not affect the final result.

1. Explain the values in the Op column for the data-processing instructions.
2. Where is ***RegWrite*** used in your datapath diagram? Explain its value in the truth table for the two kinds of data-processing instructions.
3. Where is ***ALUSrc*** used in your diagram? Explain its value in the truth table for the two kinds of data-processing instructions.

ALUSrc is used for deciding if an immediate or another register was used.

1. Where is ***MemWrite*** used in your diagram? Explain its value in the truth table for the two kinds of data-processing instructions.
2. Where is ***MemToReg*** used in your diagram? Explain its value in the truth table for the two kinds of data-processing instructions.

MemToReg is used to determine if a ldr instruction was used.

1. Where is ***ALUControl*** used in your diagram? Explain its value in the truth table for the two kinds of data-processing instructions.

ALUControl is used to determine what funct the ALU will perform.

1. Where is ***ImmSrc*** used in your diagram? Explain its value in the truth table for the two kinds of data-processing instructions.

ImmSrc is used to determine if an imm8 or an imm12 was used.

# Read This!

Putting X’s in a truth table indicate ***don’t-cares***. A don’t-care is a cell for which either 0 or 1 will work.

# Exercises

1. Fill in the table for ldr and str. Use the sequence of questions above to guide you.
2. Revisit your control signal values in Model 3. Are there any cells for which you could have picked any value (e.g., 1 or 0) and it would still be correct? (Do not change the datapath, assume it is fixed). Mark these cells with an X.
3. We are showing only some of the rows in the truth table in Model 3. How many rows are there in the full table? Explain your answer.

# Extension questions

1. What are the advantages of putting ***don’t-cares*** (X’s) in your truth table?