MUX Definitions:

```
package forwardmux;
typedef enum bit [1:0] {
    rs_out   = 2'b00,
    ex_mem_out  = 2'b01,
    mem_wb_out  = 2'b10
} forwardmux_sel_t;
endpackage
```

Forwarding Unit:
Below is the pseudo code for the forwarding unit shown in the datapath.

```
// source register 1 conflicts
forward1 = forwardmux::rs_out; // Output from register unit
if (id_ex.rs1 != 0)
     if (ex_mem.ctrl.load_regfile & (ex_mem.rd == id_ex.rs1)
          forward1 = forwardmux::ex_mem_out; // ex_mem.ex_data_out
     else if (mem_wb.ctrl.load_regfile) & (mem_wb.rd == id_ex.rs1)
          forward1 = forwardmux::mem_wb_out; // mem_wb.rd_data_in

// source register 2 conflicts  (same as above but for second mux)
forward2 = forwardmux::rs_out;
if (id_ex.rs2 != 0)
     if (ex_mem.ctrl.load_regfile) & (ex_mem.rd == id_ex.rs2)
          forward2 = forwardmux::ex_mem_out;
     else if (mem_wb.ctrl.load_regfile) & (mem_wb.rd == id_ex.rs2)
          forward2 = forwardmux::mem_wb_out;
```

Hazard Detection Unit:

```
hz_pc_ld = 1;
hz_if_id_ld = 1;
hz_if_id_rst = 0;
hz_id_ex_rst = 0;
hz_ex_mem_rst = 0;

// Data hazard
no_sr1 = [op_auipc, op_lui, op_jal] // Instr which don’t read sr1
no_sr2 = [op_auipc, op_lui, op_jal, op_load, op_imm]
```

```
if (id_ex.opcode == op_load
& id_ex.rd != 0
& ((if_id.opcode not in no_sr1) & (id_ex.rd == if_id.sr1))
| (if_id.opcode not in no_sr2) & (id_ex.rd == if_id.sr2)))
      // Stall pipeline and insert nop in ex stage
      hz_pc_ld = 0    // no load pc
      hz_if_id_ld = 0 // no load if_id
      hz_id_ex_rst = 1 // Insert nop

// Control hazard
if ((ex_mem.opcode == op_br & br_en) | ex_mem.opcode == op_jal |
ex_mem.opcode == op_jalr)
      hz_if_id_rst = 1 // Reset if_id
      hz_id_ex_rst = 1
      hz_ex_mem_rst = 1
```

Arbiter Design

The interface for the arbiter is as follows:

```
module arbiter
(
    input clk,
    input rst,

    /* Instruction memory signals */
    input   logic [31:0]    imem_address,
    output  logic [255:0]   imem_rdata,
    input   logic [255:0]   imem_wdata,
    input   logic           imem_read,
    input   logic           imem_write,
    output  logic           imem_resp,

    /* Data memory signals */
    input   logic [31:0]    dmem_address,
    output  logic [255:0]   dmem_rdata,
    input   logic [255:0]   dmem_wdata,
    input   logic           dmem_read,
    input   logic           dmem_write,
    output  logic           dmem_resp,

    /* Physical memory signals */
    output  logic [31:0]    pmem_address,
    input   logic [255:0]   pmem_rdata,
    output  logic [255:0]   pmem_wdata,
    output  logic           pmem_read,
    output  logic           pmem_write,
    input   logic           pmem_resp
);
```

The arbiter itself can be described entirely by its state machine.

Controller Description:

Default signals:

pmem_address = dmem_address;

pmem_wdata = dmem_wdata;

pmem_read = 1'b0;

pmem_write = 1'b0;

dmem_resp = 1'b0;

dmem_rdata = pmem_rdata;

imem_resp = 1'b0;

imem_rdata = pmem_rdata;

State Name	Description	Transition Conditions	Outputs

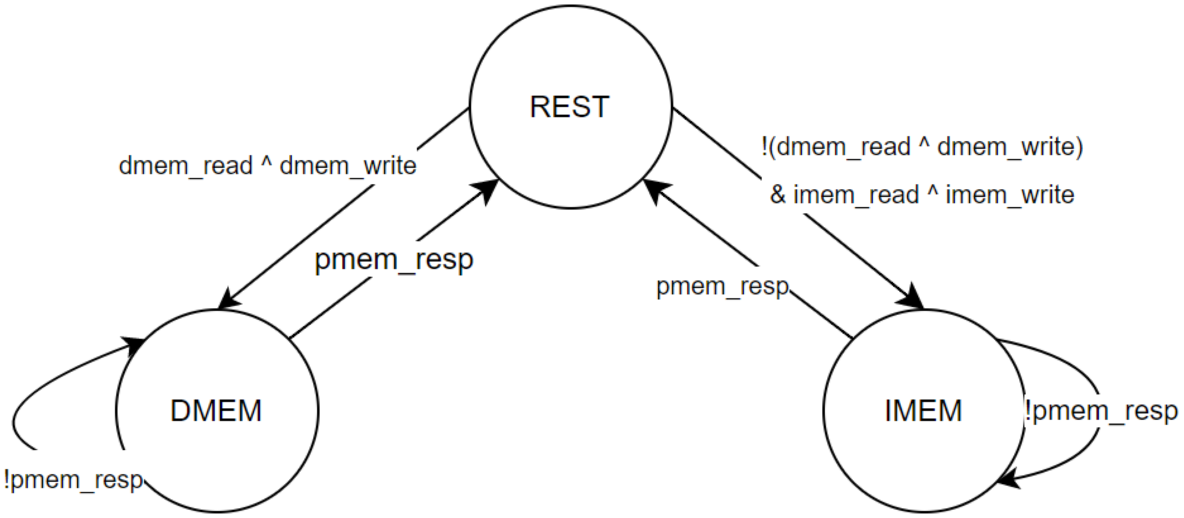
REST	Rest state. No physical memory read/writes in progress.	if (dmem_read ^ dmem_write)
		DMEM
		else if (imem_read ^ imem_write)
		IMEM
		else
		REST
			if (dmem_read ^ dmem_write)
			pmem_read = dmem_read;
			pmem_write = dmem_write;
			dmem_resp = pmem_resp;
			else if (imem_read ^ imem_write)
			pmem_read = imem_read;
			pmem_write = imem_write;
			imem_resp = pmem_resp;
			pmem_address = imem_address;
			pmem_wdata = imem_wdata;

DMEM	Accessing physical memory for data.	if (pmem_resp)
		REST
		else
		DMEM
			pmem_read = dmem_read;
			pmem_write = dmem_write;
			dmem_resp = pmem_resp;

IMEM	Accessing physical memory for instruction.	if (pmem_resp) -> REST
		else -> IMEM
			pmem_read = imem_read;
			pmem_write = imem_write;
			imem_resp = pmem_resp;
			pmem_address = imem_address;
			pmem_wdata = imem_wdata;

Note that this design assumes that main memory accesses take at least 2 cycles to complete.

State Diagram:



Arbiter Integration:

In order to integrate the multicycle memory system with the processor, the processor must be able to be stalled. In this design, whenever there is a cache miss on either the instruction or data cache, the entire pipeline will be stalled by pausing loads on the pc and the four pipeline registers. Note that these load signals will override the reset signals generated by the hazard detection unit. This stalling logic is contained in the datapath but is replicated here for clarity.

```
instr_stall = !instruction_cache.mem_resp

data_stall = (ex_mem.ctrl.mem_write | ex_mem.ctrl.mem_read) &
!data_cache.mem_resp

pipeline_stall = instr_stall | data_stall

pc.load = !pipeline_stall & hz_pc_ld
```

```
if_id.load = !pipeline_stall & hz_if_id_ld

id_ex.load = !pipeline_stall

ex_mem.load = !pipeline_stall

mem_wb.load = !pipeline_stall

if_id.rst = !pipeline_stall & hz_if_id_rst // or main reset

id_ex.rst = !pipeline_stall & hz_id_ex_rst // or main reset

ex_mem.rst = !pipeline_stall & hz_ex_mem_rst // or main reset
```

Progress Report (3/10-3/29)

Over the last two weeks, we completed the requirements for Checkpoint 1 as well as parts of checkpoint 2.

For checkpoint 1 content, the basic division of labor was as follows:

- Implementation of basic RV32I pipelined datapath: Mitchell
- Arbiter, hazard detection, and forwarding design: Neha
- Arbiter, hazard detection, and forwarding design verification: Mitchell
- Pipeline implementation verification: Yu

The basic pipelined datapath was initially tested and debugged using the mp4-cp1.s test code provided by the TA's. Once this test successfully passed, the design was further tested by writing extra assembly code, which tested each instruction independently, and inspecting the waveform to check if the outcome matches the expectation.

For checkpoint 2 content that was implemented, the division of labor was as follows:

- Integration of caches, arbiter, hazard detection, forwarding, and static branch prediction: Neha
- RVFI + verification: Yu

First, hazard detection, forwarding, and static branch prediction were implemented and tested using the magic memory from checkpoint 1. Assembly test code was written to test each of the different forwarding, data hazard, and control hazard possibilities. To ensure correct behavior of this test code, Verdi was used to trace the timing for each of the instructions. Once this part of the design was verified, the arbiter was implemented and integrated with the given cache. From here, the same testing procedure was repeated, and we ensured that the mp4-cp2.s test code also passed. Finally, once the RVFI was correctly hooked up, the test code was rerun to ensure that our design matches the specifications.

The fmax and total power for our design as of now is given below. The full timing and power reports are at the end of this document.

slack = 5.15

fmax = 1/(10-slack) * 1000 = 206.2 MHz

total power = 1.78e+03 uW

A datapath for our design up to this point is given at the end of this document.

Roadmap (3/29-4/12)

As our next step, we plan to start working on designing the advanced features. The following is a list of advanced features we are planning to look into:

Cache organization and design

L2+ cache system[2]

4-way set associative cache[2]

Parameterized cache[points up to TA discretion]

Branch prediction

Local branch history table[2]

Global 2-level branch history table[3]

Tournament branch predictor[5]

Software branch predictor model [2]

Advanced cache

Victim Cache[6]

Other than the advanced features, we are also looking for a more efficient branch stall design and adding the instruction for execution of coremark.

The work division for now is as follows:

- Cache organization and design options: Neha
- More efficient control hazard handling & victim cache: Mitchell
- Branch prediction & coremark: Yu

Each member is responsible for the design, implementation, and verification of their respective part. Our main goal for the next two weeks will be to do the design component.

ex_mem.ctrl.opcode
ex_mem.br_en
hz_pc_ld
hz_if_id_ld
hz_if_id_rst
hx_id_ex_ld
hz_id_ex_rst
hz_ex_mem_rst
Hazard Detection Unit
id_ex.rd
id_ex.ctrl.opcode
id_ex.sr1
id_ex.sr2
Forwarding Unit
ex_mem.ctrl.load_regfile
ex_mem.rd
mem_wb.ctrl.load_regfile
mem_wb.rd
rd
+4
pc
pc
rd
sr1
sr2
pc
rd
pc
rd
forwardmux1_sel
id_ex.ctrl.alumux1_sel
ex_mem.ctrl.mem_write
ex_mem.ctrl.mem_read
data_stall
mem_wb.ctrl.regfilemux_sel
mem_wb.ctrl.load_regfile
rs_out
mem_wb_out
ex_mem_out
Forward MUX 1
rs1_out
0
ALU MUX 1
1
pc
id_ex.ctrl.aluop
mem_read
mem_write
mem_resp
!instr_stall & !data_stall & hz_pc_ld
instr_stall
{ex_data_out[31:2], 2'b0}
address
read_data
Data Cache
read_data
rd_data decoder
rd_data_in
pc_plus4
alu_out
alu_mod2
0
1
2
PC MUX
load
PC
mem_resp
address
Instruction Cache
rdata
rs1
rs2
rd
rs1_out
rs2_out
Registers
rd_data_in
ALU
alu_out
Signals to/from Arbiter
forwardmux2_sel
rs2_out
0
ALU MUX 2
1
imm
Forward MUX 2
Instruction Decoder
IF/ID Register
ID/EX Register
EX/MEM Register
MEM/WB Register
MOD4
mem_write_en
write_data
ex_data_out[1:0]
addr[1:0]
pcmux_sel
mem_write_en compute
ex_data_out
id_ex.alumux2_sel
ex_mem.ctrl.funct3
Signals to/from Arbiter
rs2_out
ctrl
ctrl
ctrl
opcode
funct3
funct7
Control ROM
alu_out
0
br_en
1
u_imm
2
pc_plus4
3
EX Data MUX
ex_data_out
ex_data_out
+4
MOD2
ctrl.immmux_sel
i_imm
u_imm
b_imm
s_imm
j_imm
0
1
2
3
4
IMM MUX
imm
ZEXT
id_ex.ctrl.exdatamux_sel
br_en
rs2_out
0
{rs2_out[15:0], rs2_out[15:0]}
1
{rs2_out[7:0], rs2_out[7:0], rs2_out[7:0], rs2_out[7:0]}
2
Write Data MUX
write_data_mux_out
br_en
rs2_out
0
CMP MUX
i_imm
1
rs1_out
cmpmux_out
CMP
ex_mem.ctrl.wdata_mux_sel
ex_mem.ctrl.opcode
load
rst
load
rst
load
rst
load
!instr_stall & !data_stall & hz_if_id_ld
!instr_stall & !data_stall & hz_if_id_rst
!instr_stall & !data_stall
!instr_stall & !data_stall & hz_id_ex_rst
id_ex.ctrl.cmpmux_sel
id_ex.ctrl.cmpop
!data_stall & !instr_stall
!data_stall & !instr_stall & hx_ex_mem_rst
br_en
pcmux_sel compute
!data_stall & !instr_stall
pcmux_sel
alu_out
pcmux_sel
rd_data_in
signals to/from instruction cache
Arbiter
signals to/from data cache
signals to/from physical memory

```
Information: Updating design information... (UID-85)
Warning: There are 291 switching activity information conflicts. (PWR-19)
Information: Propagating switching activity (high effort zero delay
simulation). (PWR-6)
Warning: Design has unannotated sequential cell outputs. (PWR-415)

****************************************
Report : power
        -hier
        -analysis_effort high
Design : mp4
Version: R-2020.09-SP4
Date   : Mon Mar 27 19:37:58 2023
****************************************


Library(s) Used:

    NangateOpenCellLibrary (File: /class/ece411/freepdk-45nm/stdcells.db)


Operating Conditions: typical   Library: NangateOpenCellLibrary
Wire Load Model Mode: top

Design        Wire Load Model            Library
------------------------------------------------
mp4                    5K_hvratio_1_1    NangateOpenCellLibrary


Global Operating Voltage = 1.1
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


-------------------------------------------------------------------------
-------
                                       Switch   Int      Leak     Total
Hierarchy                              Power    Power    Power
Power    %
-------------------------------------------------------------------------
-------
mp4                                     272.791  757.822 7.49e+05
1.78e+03 100.0
  cacheline_adapter (cacheline_adaptor)
                                         20.161   45.124 3.70e+04
102.238   5.7
  arbiter (arbiter)                       5.671    4.263 9.83e+03
19.767    1.1
  dcache (cache_0)                       64.613  186.456 2.43e+05
493.621  27.7
```

```
    bus (line_adapter_0)                  0.242    0.175 6.12e+03
6.533   0.4
    datapath (cache_datapath_0)          64.307  185.684 2.36e+05
486.095  27.3
      dirty (array_width1_0)              1.341    6.092 1.60e+03
9.036   0.5
      valid (array_width1_1)              0.121    0.696 1.53e+03
2.345   0.1
      tag (array_width24_0)               0.368    5.493 1.92e+04
25.065   1.4
      DM_cache (data_array_0)            61.554  172.762 1.98e+05
432.111  24.3
    control (cache_control_0)          6.33e-02    0.598  332.251
0.993   0.1
  icache (cache_1)                       76.643  202.565 2.48e+05
526.882  29.6
    bus (line_adapter_1)                  1.285    0.641 4.24e+03
6.161   0.3
    datapath (cache_datapath_1)          75.290  201.421 2.43e+05
520.000  29.2
      valid (array_width1_3)              0.139    0.822 1.59e+03
2.553   0.1
      tag (array_width24_1)               0.979    6.006 1.94e+04
26.418   1.5
      DM_cache (data_array_1)            72.738  193.807 2.14e+05
480.348  27.0
    control (cache_control_1)          6.90e-02    0.503  149.170
0.721   0.0
  cpu (cpu)                             105.293  319.292 2.10e+05
634.864  35.7
    datapath (datapath)                 105.293  319.292 2.10e+05
634.864  35.7
      rd_data_decoder (rd_data_decoder)
                                          0.424    0.173 2.78e+03
3.378   0.2
      forwarding_unit (forwarding)        0.234    0.318 1.19e+03
1.746   0.1
      CMP (cmp)                           0.347    0.473 4.42e+03
5.234   0.3
      ALU (alu)                           0.891    0.938 2.26e+04
24.441   1.4
      hazard_detection (hazard_detection)
                                          0.141    0.223  995.991
1.360   0.1
      control_rom (control_rom)           0.518    0.247 2.41e+03
3.180   0.2
      regfile (regfile)                   6.089   29.810 1.12e+05
147.573   8.3
      instr_decoder (instr_decoder)       0.000    0.000    0.000
0.000   0.0
      PC (pc_register)                    1.476    3.919 3.24e+03
8.632   0.5
      mem_wb_reg (mem_wb_reg)            14.230   41.075 7.27e+03
62.572   3.5
```

```
      ex_mem_reg (ex_mem_reg)            27.057   80.468 1.18e+04
119.314   6.7
      id_ex_reg (id_ex_reg)              38.577  118.398 1.68e+04
173.765   9.8
      if_id_reg (if_id_reg)              14.090   42.505 6.39e+03
62.983   3.5
1
```

```
****************************************
Report : timing
        -path full
        -delay max
        -max_paths 1
Design : mp4
Version: R-2020.09-SP4
Date   : Mon Mar 27 19:36:14 2023
****************************************

Operating Conditions: typical   Library: NangateOpenCellLibrary
Wire Load Model Mode: top

  Startpoint: cpu/datapath/mem_wb_reg/data_reg[rd][0]
              (rising edge-triggered flip-flop clocked by my_clk)
  Endpoint: cpu/datapath/ex_mem_reg/data_reg[ex_data_out][31]
            (rising edge-triggered flip-flop clocked by my_clk)
  Path Group: my_clk
  Path Type: max

  Des/Clust/Port     Wire Load Model       Library
  ------------------------------------------------
  mp4                5K_hvratio_1_1        NangateOpenCellLibrary

  Point                                                    Incr       Path
  -----------------------------------------------------------------------
---
  clock my_clk (rise edge)                                 0.00       0.00
  clock network delay (ideal)                              0.00       0.00
  cpu/datapath/mem_wb_reg/data_reg[rd][0]/CK (DFF_X1)      0.00       0.00
r
  cpu/datapath/mem_wb_reg/data_reg[rd][0]/Q (DFF_X1)       0.10       0.10
f
  cpu/datapath/mem_wb_reg/out[rd][0] (mem_wb_reg)          0.00       0.10
f
  cpu/datapath/forwarding_unit/wb_rd[0] (forwarding)       0.00       0.10
f
  cpu/datapath/forwarding_unit/U20/ZN (INV_X1)             0.06       0.15
r
  cpu/datapath/forwarding_unit/U21/ZN (OAI22_X1)           0.04       0.19
f
  cpu/datapath/forwarding_unit/U22/ZN (AOI221_X1)          0.09       0.28
r
  cpu/datapath/forwarding_unit/U24/ZN (NAND4_X1)           0.05       0.33
f
  cpu/datapath/forwarding_unit/U25/ZN (NOR3_X1)            0.07       0.40
r
  cpu/datapath/forwarding_unit/forward2[1] (forwarding)
                                                           0.00       0.40
r
  cpu/datapath/U36/ZN (INV_X1)                             0.03       0.42
f
```

```
  cpu/datapath/U40/ZN (NOR2_X1)                           0.16       0.58
r
  cpu/datapath/U41/Z (CLKBUF_X1)                          0.19       0.77
r
  cpu/datapath/U42/ZN (NOR2_X1)                           0.11       0.88
f
  cpu/datapath/U43/Z (CLKBUF_X1)                          0.17       1.05
f
  cpu/datapath/U224/ZN (AOI22_X1)                         0.10       1.15
r
  cpu/datapath/U225/ZN (OAI21_X1)                         0.04       1.19
f
  cpu/datapath/U227/ZN (INV_X1)                           0.04       1.23
r
  cpu/datapath/U229/ZN (AOI22_X1)                         0.03       1.26
f
  cpu/datapath/U230/Z (CLKBUF_X1)                         0.13       1.39
f
  cpu/datapath/ALU/b[2] (alu)                             0.00       1.39
f
  cpu/datapath/ALU/U37/ZN (INV_X1)                        0.15       1.54
r
  cpu/datapath/ALU/U25/ZN (INV_X1)                        0.10       1.64
f
  cpu/datapath/ALU/U113/Z (XOR2_X1)                       0.11       1.75
f
  cpu/datapath/ALU/U186/CO (FA_X1)                        0.11       1.86
f
  cpu/datapath/ALU/U255/CO (FA_X1)                        0.09       1.95
f
  cpu/datapath/ALU/U360/CO (FA_X1)                        0.09       2.04
f
  cpu/datapath/ALU/U382/CO (FA_X1)                        0.09       2.13
f
  cpu/datapath/ALU/U400/CO (FA_X1)                        0.09       2.22
f
  cpu/datapath/ALU/U673/CO (FA_X1)                        0.09       2.31
f
  cpu/datapath/ALU/U415/CO (FA_X1)                        0.09       2.40
f
  cpu/datapath/ALU/U429/CO (FA_X1)                        0.09       2.49
f
  cpu/datapath/ALU/U450/CO (FA_X1)                        0.09       2.58
f
  cpu/datapath/ALU/U469/CO (FA_X1)                        0.09       2.67
f
  cpu/datapath/ALU/U483/CO (FA_X1)                        0.09       2.76
f
  cpu/datapath/ALU/U500/CO (FA_X1)                        0.09       2.86
f
  cpu/datapath/ALU/U519/CO (FA_X1)                        0.09       2.95
f
  cpu/datapath/ALU/U557/CO (FA_X1)                        0.09       3.04
f
```

```
  cpu/datapath/ALU/U687/CO (FA_X1)                        0.09       3.13
f
  cpu/datapath/ALU/U700/CO (FA_X1)                        0.09       3.22
f
  cpu/datapath/ALU/U713/CO (FA_X1)                        0.09       3.31
f
  cpu/datapath/ALU/U725/CO (FA_X1)                        0.09       3.40
f
  cpu/datapath/ALU/U736/CO (FA_X1)                        0.09       3.49
f
  cpu/datapath/ALU/U746/CO (FA_X1)                        0.09       3.58
f
  cpu/datapath/ALU/U763/CO (FA_X1)                        0.09       3.67
f
  cpu/datapath/ALU/U775/CO (FA_X1)                        0.09       3.76
f
  cpu/datapath/ALU/U576/CO (FA_X1)                        0.09       3.85
f
  cpu/datapath/ALU/U600/CO (FA_X1)                        0.09       3.94
f
  cpu/datapath/ALU/U786/CO (FA_X1)                        0.09       4.03
f
  cpu/datapath/ALU/U798/CO (FA_X1)                        0.09       4.12
f
  cpu/datapath/ALU/U617/CO (FA_X1)                        0.09       4.22
f
  cpu/datapath/ALU/U639/CO (FA_X1)                        0.09       4.31
f
  cpu/datapath/ALU/U660/CO (FA_X1)                        0.09       4.40
f
  cpu/datapath/ALU/U663/Z (XOR2_X1)                       0.07       4.46
f
  cpu/datapath/ALU/U664/ZN (AND2_X1)                      0.04       4.50
f
  cpu/datapath/ALU/U665/ZN (OR2_X1)                       0.06       4.56
f
  cpu/datapath/ALU/f[31] (alu)                            0.00       4.56
f
  cpu/datapath/U652/ZN (AOI22_X1)                         0.06       4.62
r
  cpu/datapath/U653/ZN (OAI21_X1)                         0.03       4.65
f
  cpu/datapath/ex_mem_reg/in[ex_data_out][31] (ex_mem_reg)
                                                          0.00       4.65
f
  cpu/datapath/ex_mem_reg/U102/ZN (AND2_X1)               0.05       4.70
f
  cpu/datapath/ex_mem_reg/data_reg[ex_data_out][31]/D (DFF_X1)
                                                          0.01       4.71
f
  data arrival time                                                  4.71

  clock my_clk (rise edge)                               10.00      10.00
  clock network delay (ideal)                             0.00      10.00
  clock uncertainty                                      -0.10       9.90
  cpu/datapath/ex_mem_reg/data_reg[ex_data_out][31]/CK (DFF_X1)
                                                          0.00       9.90
r
  library setup time                                     -0.04       9.86
  data required time                                                 9.86
  -----------------------------------------------------------------------
---
  data required time                                                 9.86
  data arrival time                                                 -4.71
  -----------------------------------------------------------------------
---
  slack (MET)                                                        5.15


1
```