



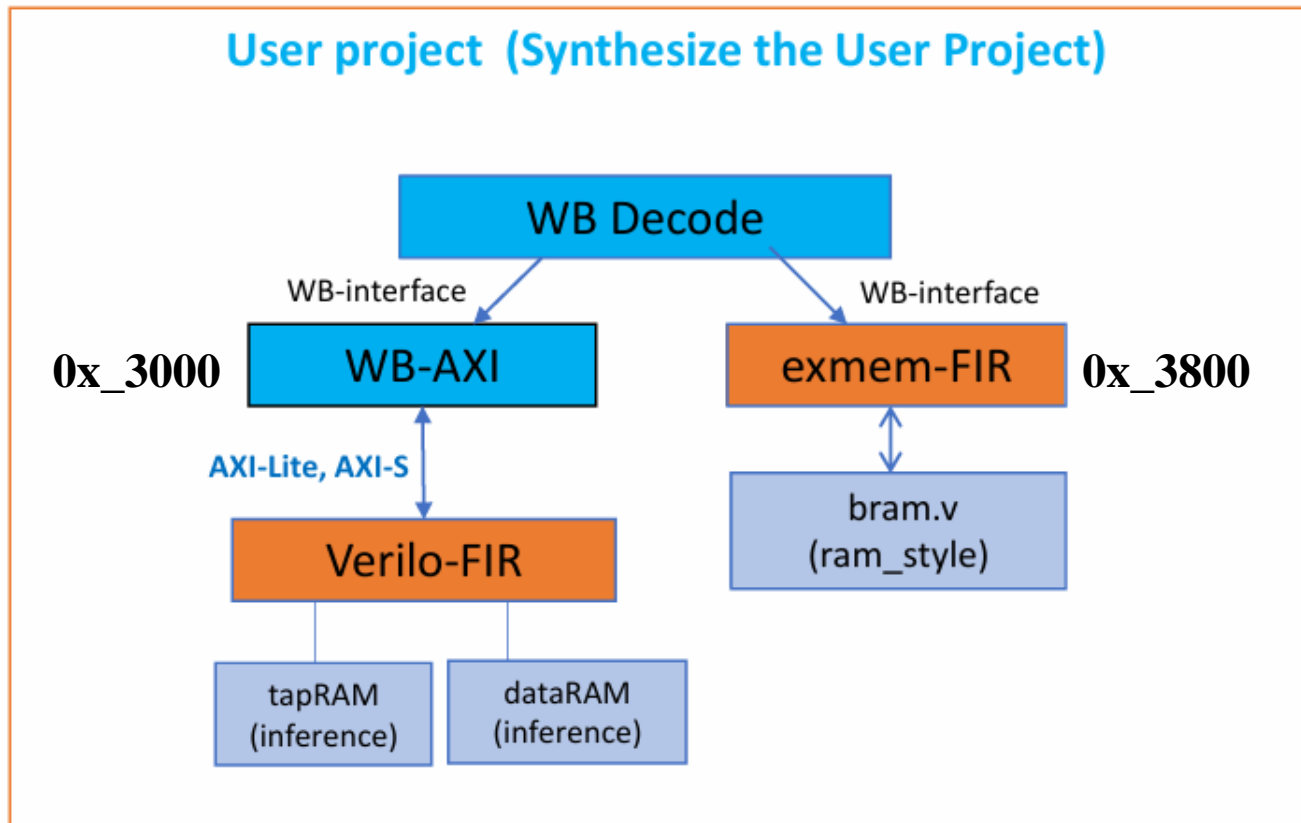
# Lab4 Caravel-FIR Hardware-Software Codesign

---

陳瀚坪 姜維 吳鎧帆

# Block Diagram

- Datapath



Module to design

RAM module provided

From previous project

# AXI read and write

```
assign arvalid = fir_addr && wbs_cyc_i && wbs_stb_i && (!wbs_we_i);
assign rready = fir_addr && wbs_cyc_i && wbs_stb_i && (!wbs_we_i);
assign araddr = wbs_adr_i;
//read ack
assign wbs_r_ack = rvalid && rready;
//read data,從output rdata中把data存到wbs_dat_o
assign wbs_r_dat = rdata;
```

```
assign awvalid = fir_addr && wbs_cyc_i && wbs_stb_i && wbs_we_i && (!stream_in) && (!stream_out);
assign wdata = wbs_dat_i;
assign wvalid = fir_addr && wbs_cyc_i && wbs_stb_i && wbs_we_i && (!stream_in) && (!stream_out);
assign awaddr = wbs_adr_i;
//write ack
assign wbs_w_ack = wready && wvalid;
//write data,把input wdata連到wbs_dat_i
assign wdata = wbs_dat_i;
```

# stream in and out

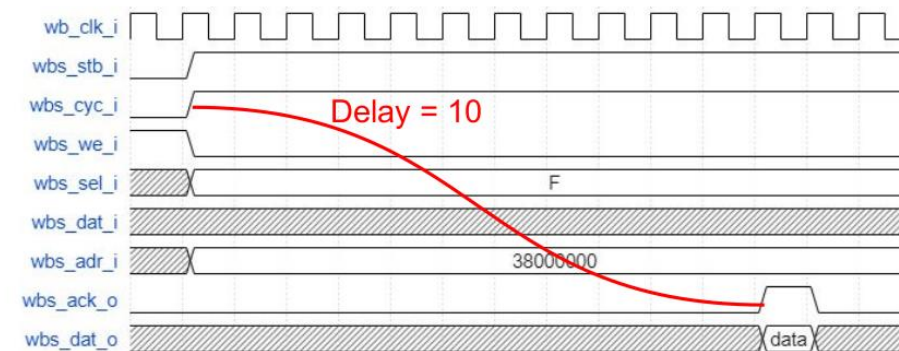
```
//-----stream in-----  
  
assign ss_tvalid = stream_in && wbs_cyc_i && wbs_stb_i ;  
assign ss_tdata = wbs_dat_i;  
//assign ss_tlast = ;  
assign wbs_si_ack = ss_tready && ss_tvalid;  
//assign wbs_si_dat = ; dont care  
  
//-----stream out-----  
  
assign sm_tready = stream_out && wbs_cyc_i && wbs_stb_i ;  
assign wbs_so_ack = sm_tvalid && sm_tready;  
assign wbs_so_dat = sm_tdata;
```

```
//-----select output-----  
always@(*) begin  
    if(fir_addr) begin  
        if(wbs_we_i) begin  
            if(wbs_adr_i[7:0] == 8'h80) begin  
                temp1 = wbs_si_ack;  
                temp2 = 32'd0;  
            end  
            else begin  
                temp1 = wbs_w_ack;  
                temp2 = 32'd0;  
            end  
        end  
        else begin  
            if(wbs_adr_i[7:0] == 8'h84) begin  
                temp1 = wbs_so_ack;  
                temp2 = wbs_so_dat;  
            end  
            else begin  
                temp1 = wbs_r_ack;  
                temp2 = wbs_r_dat;  
            end  
        end  
    end  
    //user project  
    else begin  
        temp1 = usr1;  
        temp2 = usr2;  
    end  
end  
end
```

- Exmem FIR

```
80  reg  [ 3:0] delay_cnt;
81  wire [ 3:0] bram_we   = wbs_sel_i & {4{wbs_we_i}};
82  wire      bram_en    = wbs_cyc_i & wbs_stb_i & (wbs_adr_i[31:16] == 16'h3800);
83  wire [31:0] bram_di   = wbs_dat_i;
84  wire [31:0] bram_do;
85  wire [31:0] bram_adr  = wbs_adr_i;
86  assign      wbs_dat_o = bram_do;
87  assign      wbs_ack_o = (delay_cnt == 10)? 1 : 0;
88
89  always@(posedge wb_clk_i or posedge wb_rst_i) begin
90      if(wb_rst_i) delay_cnt <= 0;
91      else if(bram_en && delay_cnt == 10) delay_cnt <= 0;
92      else if(bram_en && delay_cnt < 10) delay_cnt <= delay_cnt + 1;
93      else delay_cnt <= 0;
94  end
95
96  bram user_bram (
97      .CLK(wb_clk_i),
98      .WE0(bram_we),
99      .EN0(bram_en),
100     .Di0(bram_di),
101     .Do0(bram_do),
102     .A0(bram_adr)
103 );
```

```
input wb_clk_i,
input wb_rst_i,
input wbs_stb_i,
input wbs_cyc_i,
input wbs_we_i,
input [3:0] wbs_sel_i,
input [31:0] wbs_dat_i,
input [31:0] wbs_adr_i,
output wbs_ack_o,
output [31:0] wbs_dat_o,
```



Read BRAM



# Waveform

- Waveform (303T / per Y)
- $32' \text{ d}10614 = 32' \text{ h}2976 (76_h = 118_d)$

```
ubuntu@ubuntu2004: ~/caravel-soc_fpga-lab/lab-caravel_fir/testbench/counter_la_fir
ubuntu@ubuntu2004:~/caravel-soc_fpga-lab/lab-caravel_fir/testbench/counter_la_fir$ ./run_sim
../../rtl/user/user_project_wrapper.v:298: warning: Port 6 (A) of bram11 expects 12 bits, got 32.
../../rtl/user/user_project_wrapper.v:298:      : Pruning 20 high bits of the expression.
../../rtl/user/user_project_wrapper.v:307: warning: Port 6 (A) of bram11 expects 12 bits, got 32.
../../rtl/user/user_project_wrapper.v:307:      : Pruning 20 high bits of the expression.
Reading counter_la_fir.hex
counter_la_fir.hex loaded into memory
Memory 5 bytes = 0x6f 0x00 0x00 0x0b 0x13
VCD info: dumpfile counter_la_fir.vcd opened for output.
-----
----> Start check bits      1
----> Start FIR Test      1
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      19401 cycles
-----
----> Start check bits      2
----> Start FIR Test      2
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      19401 cycles
-----
----> Start check bits      3
----> Start FIR Test      3
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      19401 cycles
-----
Total cycles:      58203 cycles
ubuntu@ubuntu2004:~/caravel-soc_fpga-lab/lab-caravel_fir/testbench/counter_la_fir$
```

# Waveform (Instruction Reorder )

- Waveform (243T / per Y)
- $36\text{cycle}(\text{SS}) + 67\text{cycle}(\text{Fir.v}) + 140\text{cycle}(\text{SM})$

```
ubuntu@ubuntu2004:~/caravel-soc_fpga-lab/lab-caravel_fir/testbench/counter_la_fir$ ./run_sim
../../rtl/user/user_project_wrapper.v:320: warning: Port 6 (A) of bram11 expects 12 bits, got 32.
../../rtl/user/user_project_wrapper.v:320:      : Pruning 20 high bits of the expression.
../../rtl/user/user_project_wrapper.v:331: warning: Port 6 (A) of bram11 expects 12 bits, got 32.
../../rtl/user/user_project_wrapper.v:331:      : Pruning 20 high bits of the expression.
Reading counter_la_fir.hex
counter_la_fir.hex loaded into memory
Memory 5 bytes = 0x6f 0x00 0x00 0x0b 0x13
VCD info: dumpfile counter_la_fir.vcd opened for output.
-----
----> Start check bits      1
----> Start FIR Test      1
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      16034 cycles
-----
----> Start check bits      2
----> Start FIR Test      2
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      16034 cycles
-----
----> Start check bits      3
----> Start FIR Test      3
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      16034 cycles
-----
Total cycles:      48102 cycles
```



# Waveform (Instruction Reorder & -O2)

- Waveform (23T / per Y)
- 9cycle(SS) + 14cycle(Fir.v)

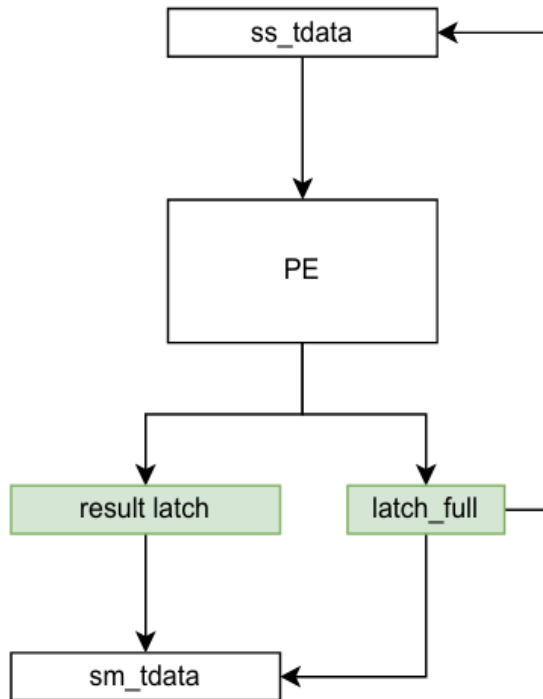
```
ubuntu@ubuntu2004:~/caravel-soc_fpga-lab/lab-caravel_fir/testbench/counter_la_fir$ ./run_sim
../../rtl/user/user_project_wrapper.v:320: warning: Port 6 (A) of bram11 expects 12 bits, got 32.
../../rtl/user/user_project_wrapper.v:320:      : Pruning 20 high bits of the expression.
../../rtl/user/user_project_wrapper.v:331: warning: Port 6 (A) of bram11 expects 12 bits, got 32.
../../rtl/user/user_project_wrapper.v:331:      : Pruning 20 high bits of the expression.
Reading counter_la_fir.hex
counter_la_fir.hex loaded into memory
Memory 5 bytes = 0x6f 0x00 0x00 0x0b 0x13
VCD info: dumpfile counter_la_fir.vcd opened for output.
-----
----> Start check bits          1
----> Start FIR Test           1
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      1700 cycles
-----
----> Start check bits          2
----> Start FIR Test           2
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      1700 cycles
-----
----> Start check bits          3
----> Start FIR Test           3
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      1700 cycles
-----
Total cycles:      5100 cycles
```

# Waveform (Instruction Reorder 2 & -O2)

- Waveform (16T / per Y)
- 2cycle(SS) + 14cycle(Fir.v)

```
ubuntu@ubuntu2004:~/caravel-soc_fpga-lab/lab-caravel_fir/testbench/counter_la_fir$ ./run_sim
../../rtl/user/user_project_wrapper.v:320: warning: Port 6 (A) of bram11 expects 12 bits, got 32.
../../rtl/user/user_project_wrapper.v:320:      : Pruning 20 high bits of the expression.
../../rtl/user/user_project_wrapper.v:331: warning: Port 6 (A) of bram11 expects 12 bits, got 32.
../../rtl/user/user_project_wrapper.v:331:      : Pruning 20 high bits of the expression.
Reading counter_la_fir.hex
counter_la_fir.hex loaded into memory
Memory 5 bytes = 0x6f 0x00 0x00 0x0b 0x13
VCD info: dumpfile counter_la_fir.vcd opened for output.
-----
----> Start check bits          1
----> Start FIR Test            1
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      1370 cycles
-----
----> Start check bits          2
----> Start FIR Test            2
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      1370 cycles
-----
----> Start check bits          3
----> Start FIR Test            3
Start latency-timer...
Final Y[7:0] = 118
Finish processing...
Executes in      1370 cycles
-----
Total cycles:      4110 cycles
```

## Improve FIR



### full - ss\_tready

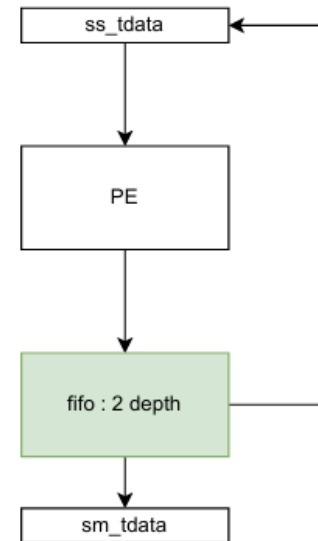
ss\_tready = !latch\_full

pe result latch , latch full <= 1

sm\_tvalid & sm\_tready & latch\_full , latch full <= 0



next ss\_tdata need to wait current sm\_tvalid



### full - ss\_tready

have 2 depth fifo can pass next ss\_tdata ready before current sm\_tdata valid.

So, it can do that, change the next x input before the current y output.

```

25  wb_write(reg_fir_x_in, t);
26  for (uint8_t t = 1; t < N; t++) {
27      temp = wb_read(reg_fir_y_out);
28      wb_write(reg_fir_x_in, t);
29      outputsignal[t - 1] = temp;
30  }
    
```



```

25  wb_write(reg_fir_x_in, t);
26  for (uint8_t t = 1; t < N; t++) {
27      wb_write(reg_fir_x_in, t);
28      temp = wb_read(reg_fir_y_out);
29      outputsignal[t - 1] = temp;
30  }
    
```

```

.L6:
    lw  a2,132(a3)
    sw  a5,128(a3)
    addi a5,a5,1
    sw  a2,0(a4)    # store to array !!!
    addi a4,a4,4
    
```

lw can't after sw x and sw array



```

.L6:
    sw  a5,128(a3)
    addi a5,a5,1
    sw  a2,0(a4)    # store to array !!!
    addi a4,a4,4
    lw  a2,132(a3)
    
```

it will use x operation cycles(11 cycles) to do these thing

# Why it can reduce cycle ?

Why it can reduce cycle ?

ss_tready	1	...	0	0	1
sm_tvalid	0	...	1	0	0

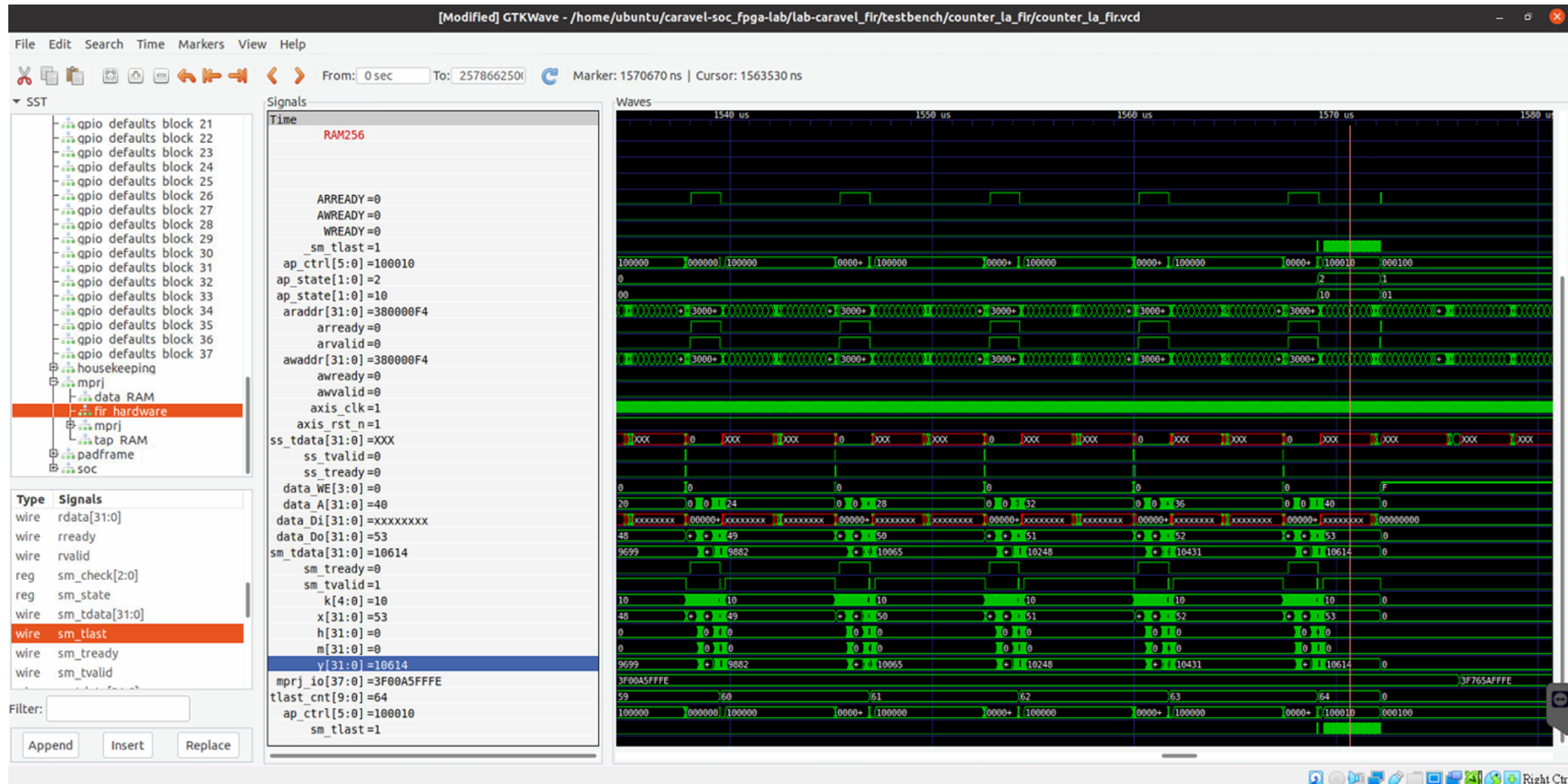
next x need wait y valid .

ss_tready	1	...	1	0	0
sm_tvalid	0	...			

next x input

current y will use these cycle to output

- Waveform





# Waveform (Optimization)

Signals

```
Time

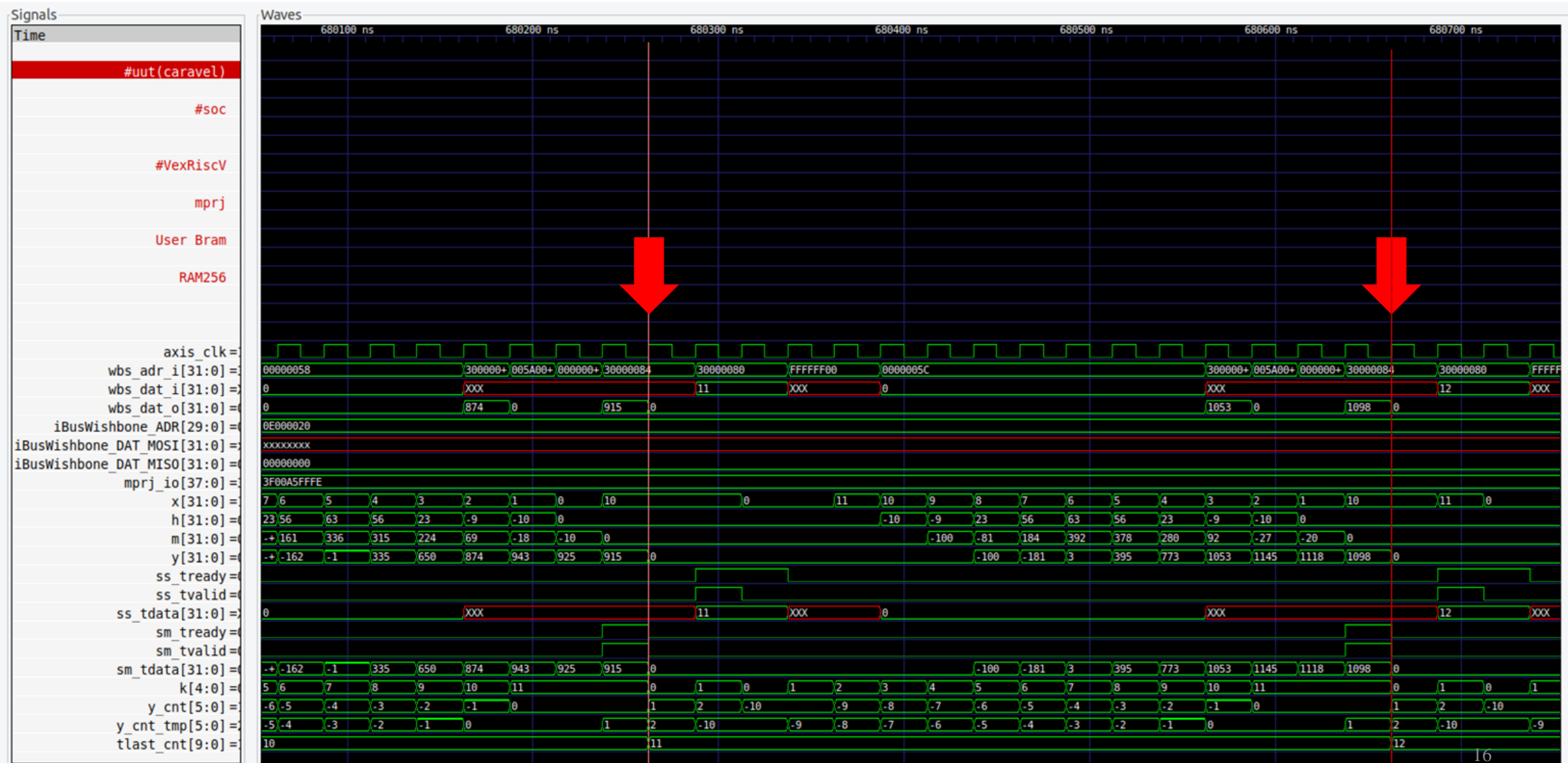
mproj
User Bram
RAM256

axis_clk=1
iBusWishbone_ADR[29:0]=0E000020
iBusWishbone_DAT_MISO[31:0]=00000000
iBusWishbone_DAT_MISO_regNext[31:0]=00000393
wbs_adr_i[31:0]=30000084
wbs_dat_i[31:0]=XXX
wbs_dat_o[31:0]=0
mproj_io[37:0]=3F00A5FFFE
x[31:0]=10
h[31:0]=0
m[31:0]=0
y[31:0]=0
ss_tready=0
ss_tvalid=0
ss_tdata[31:0]=XXX
sm_tready=0
sm_tvalid=0
sm_tdata[31:0]=0
y_cnt[5:0]=1
tap_A[31:0]=0
tap_Di[31:0]=XXX
tap_Do[31:0]=0
data_A[31:0]=0
data_Di[31:0]=XXX
data_Do[31:0]=10
data_WE[3:0]=0
data_ff[31:0]=10
ap_ctrl[5:0]=000000
tlast_cnt[9:0]=11
```

Waves



# Waveform (Optimization 2)



# Fir.c (Optimization)

```
1 #include "fir.h"
2 #include <stdint.h>
3
4 void __attribute__ ( ( section ( ".mprjram" ) ) ) initfir() {
5
6     // program data length
7     reg_fir_len = data_length;
8
9     // program coefficient
10    for (uint32_t i = 0; i < 11; i++) {
11        wb_write(reg_fir_coeff + 4*i , taps[i]);
12    }
13 }
14
15 int* __attribute__ ( ( section ( ".mprjram" ) ) ) fir_excute() {
16
17     // StartMark
18     reg_mprj_datal = 0x00A50000;
19
20     // ap_start
21     reg_fir_ap_ctrl = 1;
22
23     uint8_t register t = 0;
24
25     while (t < data_length) {
26         reg_fir_x_in = t;
27         outputsignal[t] = reg_fir_y_out;
28         t = t + 1;
29     }
30
31     // check the final Y by using MPRJ[31:24]
32     // send the EndMark 5A signal at MPRJ[23:16]
33     reg_fir_ap_ctrl; // check ap_done
34     checkbits = outputsignal[N-1] << 24 | 0x005A0000;
35
36     return outputsignal;
37 }
```

```
1 #include "fir.h"
2 #include <stdint.h>
3
4 void __attribute__ ( ( section ( ".mprjram" ) ) ) initfir() {
5
6     // program data length
7     reg_fir_len = data_length;
8
9     // program coefficient
10    for (uint32_t i = 0; i < 11; i++) {
11        wb_write(reg_fir_coeff + 4*i , taps[i]);
12    }
13 }
14
15 int* __attribute__ ( ( section ( ".mprjram" ) ) ) fir_excute() {
16
17     // StartMark
18     reg_mprj_datal = 0x00A50000;
19
20     // ap_start
21     reg_fir_ap_ctrl = 1;
22
23     uint8_t register t = 0;
24
25     //while (t < data_length) {
26     //    reg_fir_x_in = t;
27     //    outputsignal[t] = reg_fir_y_out;
28     //    t = t + 1;
29     //}
30
31     reg_fir_x_in = t;
32     while (t < data_length - 1) {
33         outputsignal[t] = reg_fir_y_out;
34         t = t + 1;
35         reg_fir_x_in = t;
36     }
37     outputsignal[t] = reg_fir_y_out;
38     // check the final Y by using MPRJ[31:24]
39     // send the EndMark 5A signal at MPRJ[23:16]
40     reg_fir_ap_ctrl; // check ap_done
41     checkbits = outputsignal[N-1] << 24 | 0x005A0000;
42
43     return outputsignal;
44 }
```

```
15 int* __attribute__ ( ( section ( ".mprjram" ) ) ) fir_excute() {
16
17     // StartMark
18     reg_mprj_datal = 0x00A50000;
19
20     // ap_start
21     reg_fir_ap_ctrl = 1;
22
23     uint8_t register t = 0;
24     uint8_t register tmp = 0;
25
26     //while (t < data_length) {
27     //    reg_fir_x_in = t;
28     //    outputsignal[t] = reg_fir_y_out;
29     //    t = t + 1;
30     //}
31
32     //reg_fir_x_in = t;
33     //while (t < data_length - 1) {
34     //    outputsignal[t] = reg_fir_y_out;
35     //    t = t + 1;
36     //    reg_fir_x_in = t;
37     //}
38     //outputsignal[t] = reg_fir_y_out;
39
40     reg_fir_x_in = t;
41     while (t < data_length - 1) {
42         tmp = reg_fir_y_out;
43         t = t + 1;
44         reg_fir_x_in = t;
45         outputsignal[t - 1] = tmp;
46     }
47
48     outputsignal[t] = reg_fir_y_out;
49
50     // check the final Y by using MPRJ[31:24]
51     // send the EndMark 5A signal at MPRJ[23:16]
52     reg_fir_ap_ctrl; // check ap_done
53     checkbits = outputsignal[N-1] << 24 | 0x005A0000;
54
55     return outputsignal;
56 }
```



```
1 #ifndef __FIR_H__
2 #define __FIR_H__
3
4 #define reg_fir_ap_ctrl (*(volatile uint32_t*)0x30000000) // ap_control
5 #define reg_fir_ss_ready (*(volatile uint32_t*)0x30000004)
6 #define reg_fir_sm_valid (*(volatile uint32_t*)0x30000008)
7
8 #define reg_fir_len (*(volatile uint32_t*)0x30000010) // data length
9 #define reg_fir_coeff 0x30000040 // Load into TapRAM
10 #define reg_fir_x_in (*(volatile uint32_t*)0x30000080) // Load X into DataRAM
11 #define reg_fir_y_out (*(volatile uint32_t*)0x30000084) // Take the output Y
12
13 #define checkbits (*(volatile uint32_t*)0x2600000C) // MPRJ I/O
14 #define reg_mprj_xfer (*(volatile uint32_t*)0x26000000)
15 #define reg_mprj_data (*(volatile uint32_t*)0x2600000c)
16
17 #define N 64
18 #define data_length N
19
20 int taps[11] = {0,-10,-9,23,56,63,56,23,-9,-10,0};
21 int outputsignal[N];
22
23 #define adr_ofst(target, offset) (target + offset)
24 #define wb_write(target, data) (*(volatile uint32_t*)(target)) = data // wishbone write
25 #define wb_read(target) (*(volatile uint32_t*)(target)) // wishbone read
26
27
28 #endif
```

# Fir\_control.c

```
1 #include <defs.h>
2 #include <stub.c>
3
4 extern int* initfir();
5 extern int* fir_excute();
6
7 void __attribute__ ( ( section ( ".mprjram" ) ) ) main()
8 {
9
10     // Step 1. Initialization code
11     reg_mprj_io_31 = GPIO_MODE_MGMT_STD_OUTPUT;
12     reg_mprj_io_30 = GPIO_MODE_MGMT_STD_OUTPUT;
13     reg_mprj_io_29 = GPIO_MODE_MGMT_STD_OUTPUT;
14     reg_mprj_io_28 = GPIO_MODE_MGMT_STD_OUTPUT;
15     reg_mprj_io_27 = GPIO_MODE_MGMT_STD_OUTPUT;
16     reg_mprj_io_26 = GPIO_MODE_MGMT_STD_OUTPUT;
17     reg_mprj_io_25 = GPIO_MODE_MGMT_STD_OUTPUT;
18     reg_mprj_io_24 = GPIO_MODE_MGMT_STD_OUTPUT;
19     reg_mprj_io_23 = GPIO_MODE_MGMT_STD_OUTPUT;
20     reg_mprj_io_22 = GPIO_MODE_MGMT_STD_OUTPUT;
21     reg_mprj_io_21 = GPIO_MODE_MGMT_STD_OUTPUT;
22     reg_mprj_io_20 = GPIO_MODE_MGMT_STD_OUTPUT;
23     reg_mprj_io_19 = GPIO_MODE_MGMT_STD_OUTPUT;
24     reg_mprj_io_18 = GPIO_MODE_MGMT_STD_OUTPUT;
25     reg_mprj_io_17 = GPIO_MODE_MGMT_STD_OUTPUT;
26     reg_mprj_io_16 = GPIO_MODE_MGMT_STD_OUTPUT;
27
28     // After setting MPRJ I/O, apply the configuration
29     reg_mprj_xfer = 1;
30     while (reg_mprj_xfer == 1);
31
32     initfir();
33
34     for (int i = 0; i < 3; i++) {
35         fir_excute();
36     }
37
38 }
```

```
Open  defs.h
~/caravel-soc_fpga-lab/lab-caravel_fir/firmware

18 #ifndef _DEF_H
19 #define _DEF_H
20
21 #include <stdint.h>
22 #include <stdbool.h>
23
24 #include <csr.h>
25 #include <caravel.h>
26
27 // a pointer to this is a null pointer, but the compiler does not
28 // know that because "sram" is a linker symbol from sections.lds.
29 extern uint32_t sram;
30
31 // Pointer to firmware flash routines
32 extern uint32_t flashio_worker_begin;
33 extern uint32_t flashio_worker_end;
34
35 // Storage area (MGMT: 0x0100_0000, User: 0x0200_0000)
36 #define reg_rw_block0 (*(volatile uint32_t*)0x01000000)
37 #define reg_rw_block1 (*(volatile uint32_t*)0x01100000)
38 #define reg_ro_block0 (*(volatile uint32_t*)0x02000000)
39
40 // UART (0x2000_0000)
41 // #define reg_uart_clkdiv (*(volatile uint32_t*)0x20000000)
42 #define reg_uart_data (*(volatile uint32_t*) CSR_UART_RXTX_ADDR)
43 #define reg_uart_txfull (*(volatile uint32_t*) CSR_UART_TXFULL_ADDR)
44 #define reg_uart_enable (*(volatile uint32_t*) CSR_UART_ENABLED_OUT_ADDR)
45 #define reg_uart_irq_en (*(volatile uint32_t*) CSR_UART_EV_ENABLE_ADDR)
46
47 // DEBUG (0x2000_0000)
48 // #define reg_uart_clkdiv (*(volatile uint32_t*)0x20000000)
49 #define reg_reset (*(volatile uint32_t*) CSR_CTRL_RESET_ADDR)
50 #define reg_debug_data (*(volatile uint32_t*) CSR_DEBUG_RXTX_ADDR)
51 #define reg_debug_txfull (*(volatile uint32_t*) CSR_DEBUG_TXFULL_ADDR)
52 #define reg_debug_irq_en (*(volatile uint32_t*) CSR_USER_IRQ_3_EV_ENABLE_ADDR)
53 // #define reg_debug_enable (*(volatile uint32_t*) CSR_UART_ENABLED_OUT_ADDR)
```

```
Open  stub.c
~/caravel-soc_fpga-lab/lab-caravel_fir/firmware

1 /*
2  * SPDX-FileCopyrightText: 2020 Efabless Corporation
3  *
4  * Licensed under the Apache License, Version 2.0 (the "License");
5  * you may not use this file except in compliance with the License.
6  * You may obtain a copy of the License at
7  *
8  * http://www.apache.org/licenses/LICENSE-2.0
9  *
10  * Unless required by applicable law or agreed to in writing, software
11  * distributed under the License is distributed on an "AS IS" BASIS,
12  * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13  * See the license for the specific language governing permissions and
14  * limitations under the License.
15  * SPDX-License-Identifier: Apache-2.0
16  */
17
18 void putchar(char c)
19 {
20     if (c == '\n')
21         putchar('\r');
22     while (reg_uart_txfull == 1);
23     reg_uart_data = c;
24 }
25
26 void print(const char *p)
27 {
28     while (*p)
29         putchar(*(p++));
30 }
```