2023年度（令和5年）版

Ver. 2023-10-24a

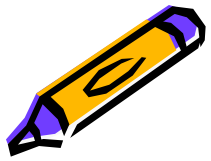
Course number: CSC.T363

コンピュータアーキテクチャ 演習(3)
Computer Architecture Exercise(3)

情報工学系　吉瀬謙二, Berjab Nesrine

Kenji KISE, Department of Computer Science

kise_at_c.titech.ac.jp



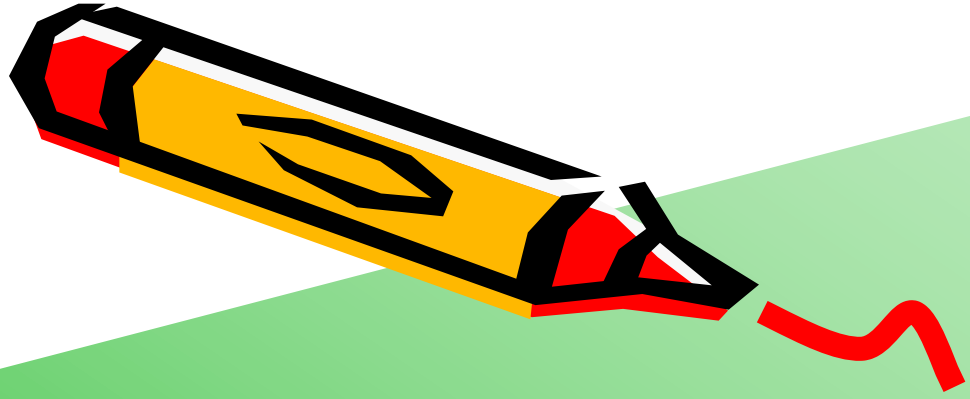
コンピュータアーキテクチャ 演習(Exercise)の注意点

- 連絡手段は Slack を利用します.
- 演習は 15:25～17:05 です.
  - 20分以上(15:45までに)遅刻したら欠席扱いになります.
  - 前半は課題の説明(15分程度)で, 後半は課題の解決とチェックポイントの確認.
- 演習は手元の FPGA ボードと ACRi ルームを利用します.
- 3~4人のグループを作成します. そのグループ内で情報を共有しながら演習を進めください.
- 問題はグループ内で相談して解決する, あるいは, 担当 TA や教員に質問してください.
- 演習には出席点があります. 休まずにきちんと出席しましょう.
- 演習スライドにチェックポイントの図がある場所は, 作業を確認してもらう場所です. すべてのチェックポイントをクリアしましょう.

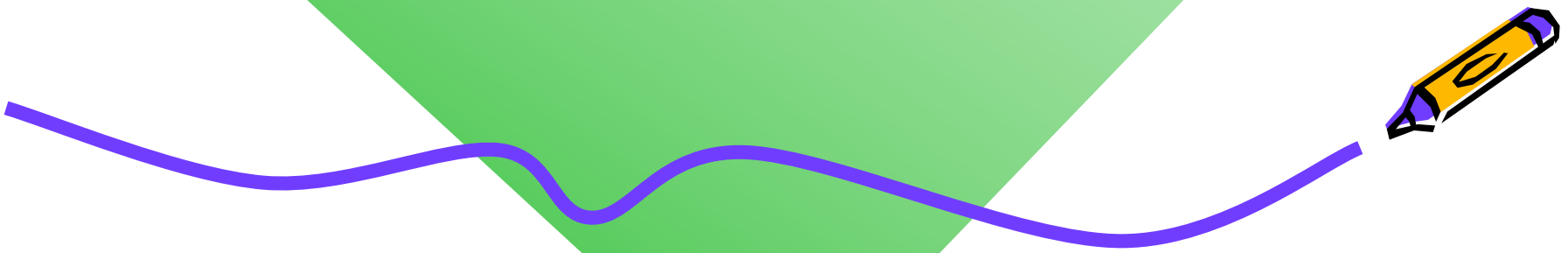
- 演習時間以外も手元の FPGA ボードと ACRi ルームを利用できます.
  - 手元の FPGA ボードを借りることができます.
  - 独自のハードウェア設計などに挑戦しましょう.

【重要】ACRiルームのサーバの予約

- ACRiルームのアカウントを使って，次のURLからログインする．
  - https://gw.acri.c.titech.ac.jp/wp
- 「予約ページトップ」から，vsで始まるサーバで演習の日の15:00～18:00の枠を予約すること．

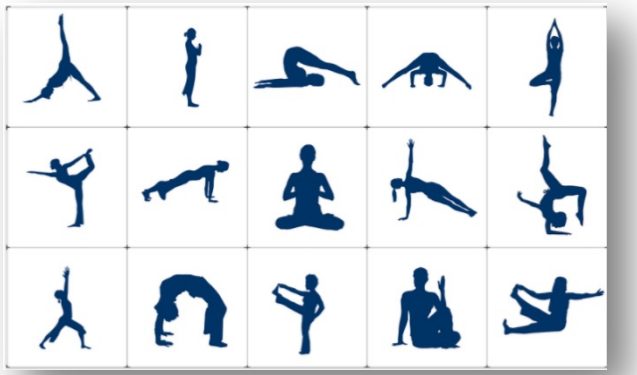


Project_22



Exercise(3)

- Project_22
  - Baseline プロセッサ (code210.v) を修正して，シリアル通信で与えられたプログラム (次のスライド)を送信することで命令メモリを初期化してから，送信したプログラムを実行するように修正する.



RISC-V Program: 1～100の合計値

- 1～100 までの合計値 (5050) を出力するプログラム.

```
#include <stdio.h>
  int main() {
    int sum = 0;
    for (int i = 1; i <= 100; i++) {
         sum += i;
    }

     printf("0000_%x\n", sum);

     return 0;
}
```

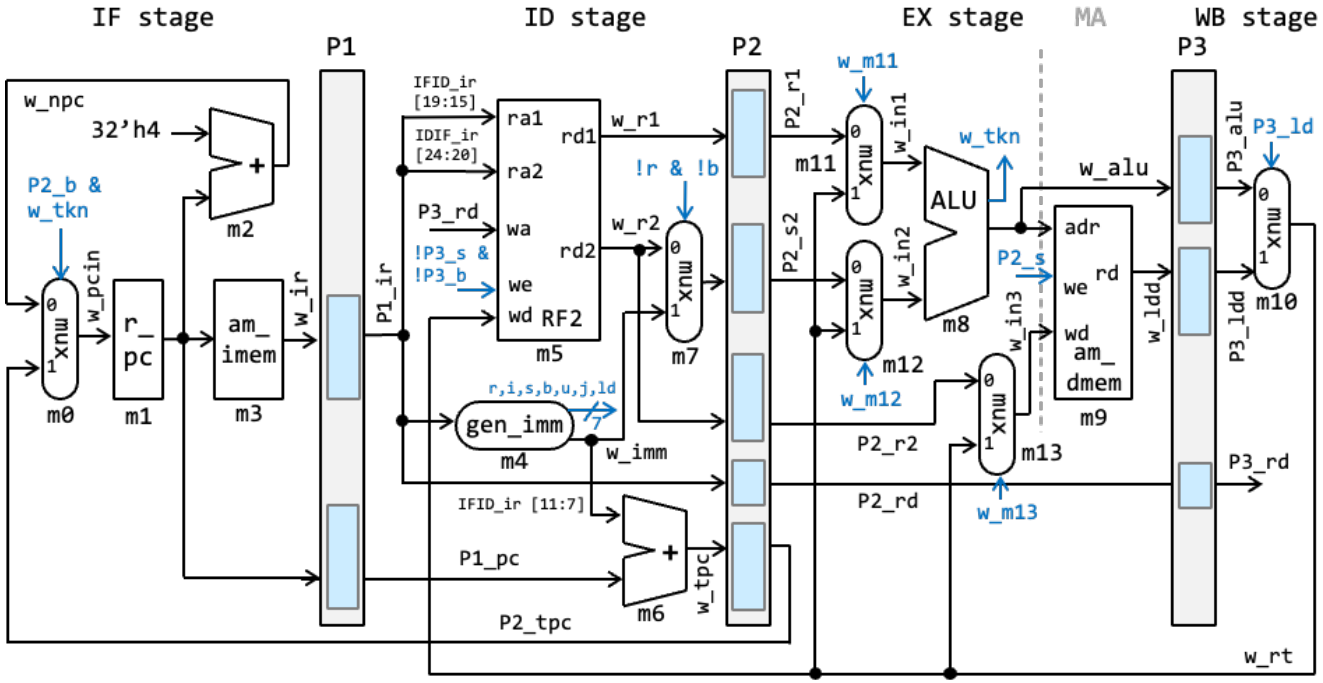
Result: 0000_13BA
(10進数で 5050)

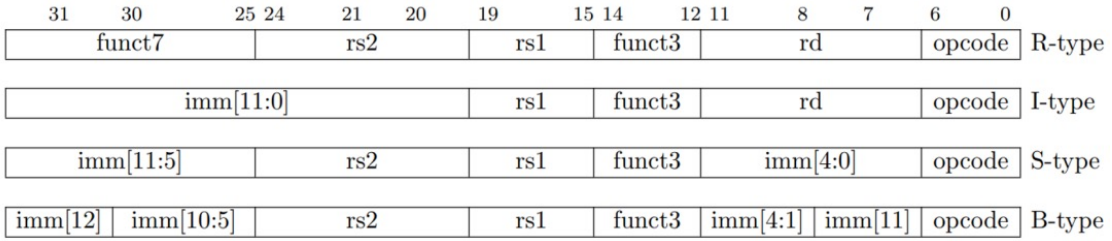
asm_program.txt

```
Initial begin
`MM[0]={12'd0,5'd0,3'h0,5'd10,7'h13}; // 00 addi x10,x0,0
`MM[1]={12'd0,5'd0,3'h0,5'd3,7'h13}; // 04 addi x3,x0,0
`MM[2]={12'd101,5'd0,3'h0,5'd1,7'h13}; // 08 addi x1,x0,101
`MM[3]={7'd0,5'd3,5'd10,3'h0,5'd10,7'h33}; // 0c L:add x10,x10,x3
`MM[4]={12'd1,5'd3,3'h0,5'd3,7'h13}; // 10 addi x3,x3,1
`MM[5]={~12'd0,5'd1,5'd3,3'h1,5'b11001,7'h63};// 14 bne x3,x1,L
`MM[6]=32'h00050f13; // 18 HALT
end
```

Inside module m_proc8s

- 4段のパイプライン処理のプロセッサ
  - 命令フェッチ(IF), デコードとオペランドフェッチ(ID), 実行(EX), メモリアクセス(MEM), ライト バック(WB) の処理をおこなうadd, addi, sll, srl, lw, sw, beq, bne命令に対応したプロセッサ





Baseline プロセッサ (code210.v) の修正

- Baselineプロセッサの命令メモリの内容を出力する.
  - memory_image.bin にバイナリ形式でメモリの内容が出力される.
  - code205.v と同じディレクトリに asm_program.txt も必要.

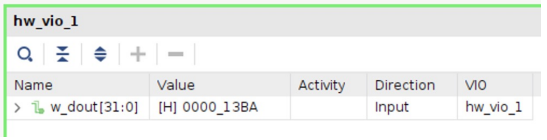
code205.v

```
module m_top ();
   reg r_clk=0; initial #150 forever #50 r_clk = ~r_clk;
   reg [31:0] r_cc=1; always @(posedge r_clk) r_cc <= r_cc + 1;

   m_proc8s m (r_clk);

   integer fp;
   integer i;

   initial begin
     `define MM m.m3.mem
     `include "asm_program.txt"
   end

   initial begin
      fp = $fopen("memory_image.bin", "wb");
      for(i=0; i<100; i=i+1) begin
         $fwrite(fp, "%u", m.m3.mem[i]);
      end
      $fclose(fp);
      $finish();
   end
endmodule
```

```
$ iverilog code205.v
$ ./a.out
$ od -tx1 memory_image.bin | head -n 2
0000000 13 05 00 00 93 01 00 00 93 00 50 06 33 05 35 00
0000020 93 81 11 00 e3 9c 11 fe 13 0f 05 00 00 00 00 00

$ od -tx memory_image.bin | head -n 2
0000000 00000513 00000193 06500093 00350533
0000020 00118193 fe119ce3 00050f13 00000000
```

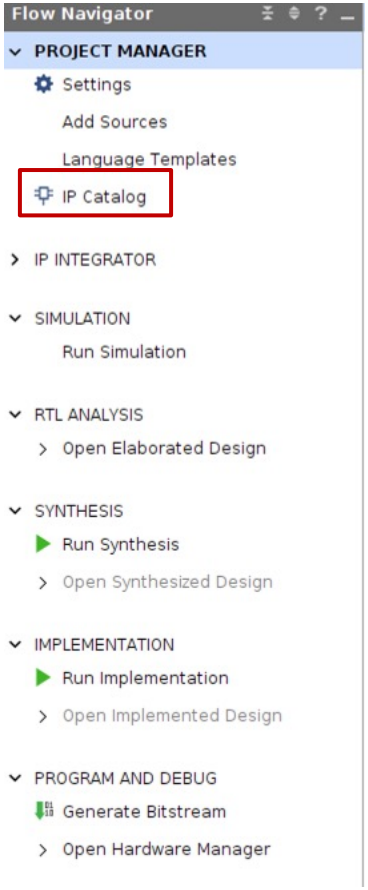
Baseline プロセッサ (code205.v) の修正

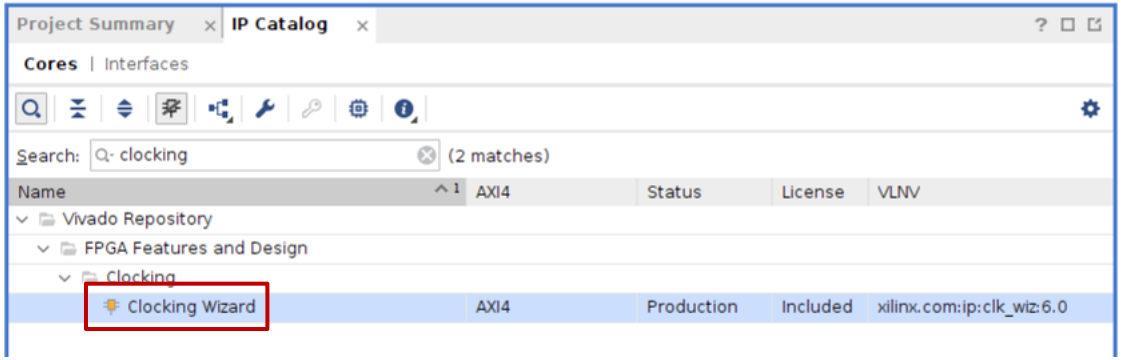
- Baseline プロセッサ (code210.v) を修正して，**シリアル通信で与えられたプログラムを送信することで命令メモリを初期化**してから，**送信したプログラム memory_image.binを実行する**ように修正する．
  - 修正した code210.v と main20.xdc をプロジェクトに追加する．
  - clocking wizard と vio をインスタンス化する．
    - 100MHz のシステムクロックを用いること．
    - vio は入力ポート数を 1、ポート幅を 32 に設定する．
- ターミナルから次のコマンドを実行．
    - cat memory_image.bin > /dev/ttyUSB1
- VIO を用いて，結果が 0x000013BA となることを確認すること．
- ポイント
  - 100MHzのシステムクロックで，先の1～100 までの合計値のプログラムの結果が正しいことを確認する．（100MHz でUARTが正しく動いていることを確認する）
  - 初期化が終わるまでプロセッサの処理をストールさせる．
  - UARTで受信したデータによって命令メモリの内容を初期化する．
  - 初期化が終わってから，プロセッサの処理を開始する．



Clocking Wizard で clock を変化させる

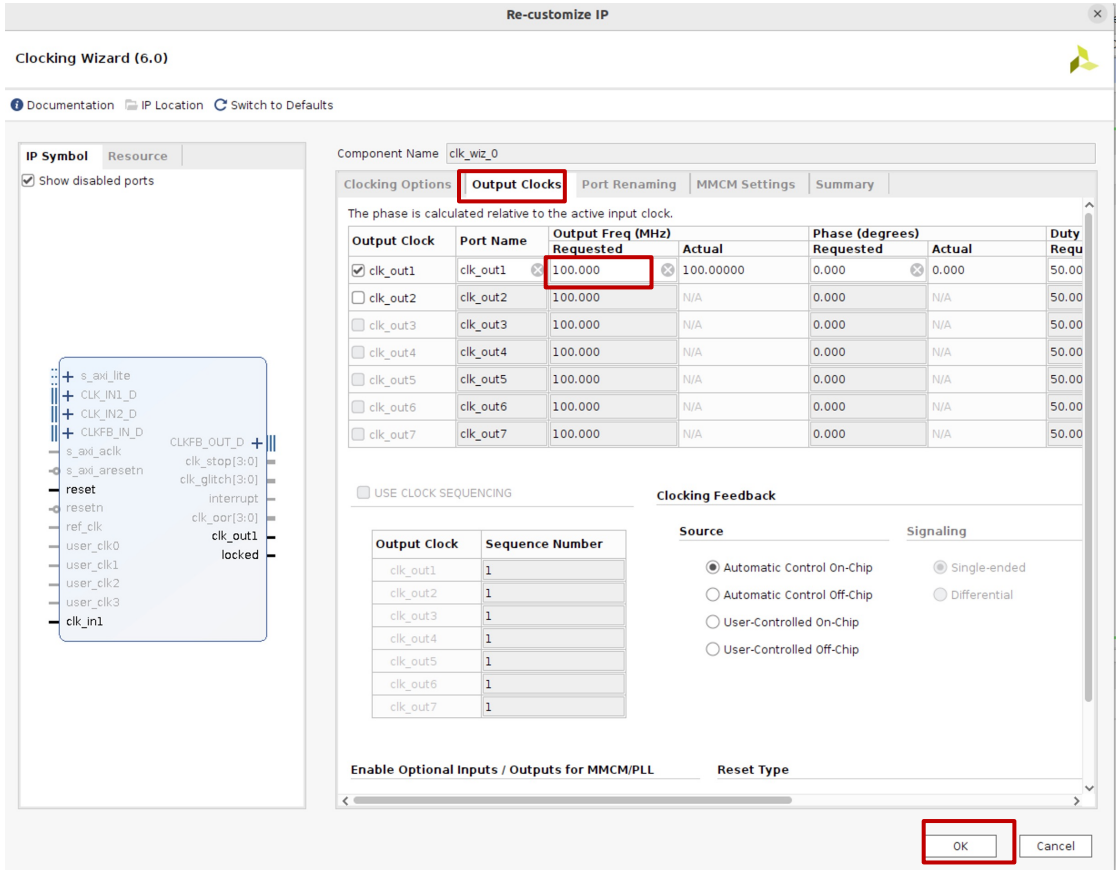
- Click IP Catalog
- Double click Clocking Wizard in IP Catalog window

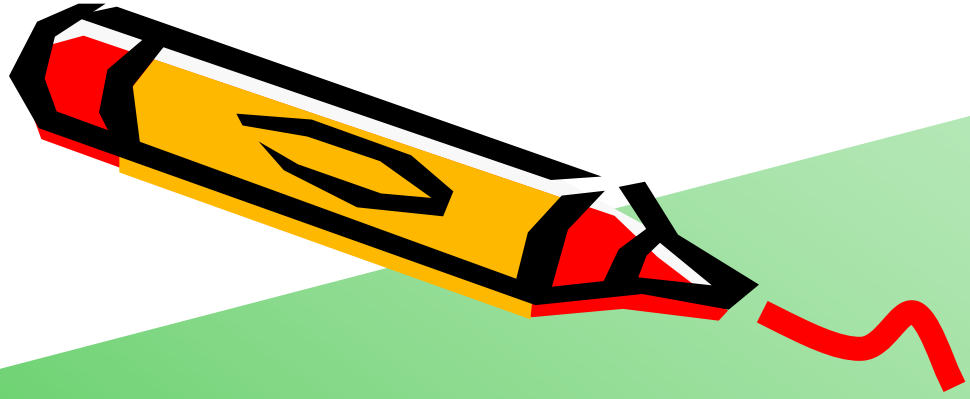




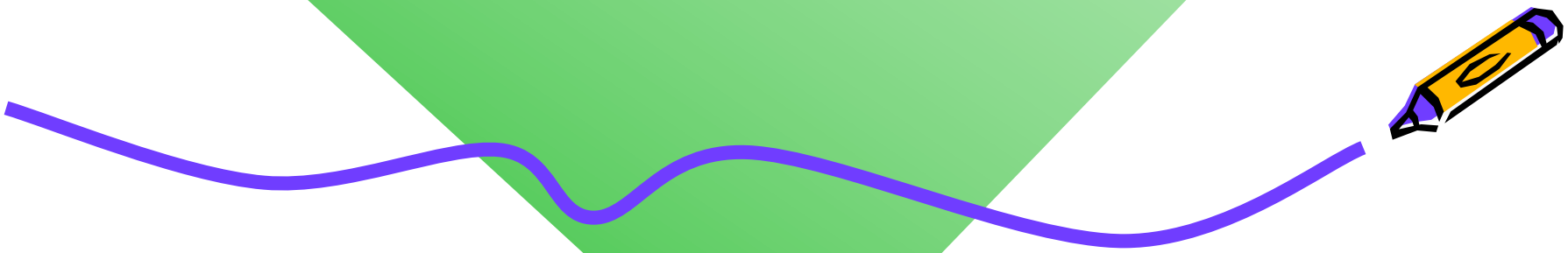
100MHzのクロックを出力する IP を生成する

- In Output Clocks, set the frequency to 100.000 for clk_out1 to generate 100MHz clock signal. Click OK.
- In Generate Output Products window, click Generate.





References



References (1/2)

- Computer Architecture support page
  - http://www.arch.cs.titech.ac.jp/lecture/CA/
- Computer Logic Design support page
  - http://www.arch.cs.titech.ac.jp/lecture/CLD/
- ACRi Room
  - https://gw.acri.c.titech.ac.jp
- ACRi Blog
  - https://www.acri.c.titech.ac.jp/wordpress/
- 情報工学系計算機室
  - http://www.csc.titech.ac.jp/

References (2/2)

- Xilinx Vivado Design Suite
  - https://japan.xilinx.com/products/design-tools/vivado.html
- Digilent Arty A7-35 A7: FPGA Trainer Board
  - https://reference.digilentinc.com/reference/programmable-logic/arty-a7/start
- Digilent Nexys 4 DDR Atrix-7 FPGA
  - https://store.digilentinc.com/nexys-4-ddr-artix-7-fpga-trainer-board-recommended-for-ece-curriculum/
- Verilog HDL
  - https://ja.wikipedia.org/wiki/Verilog