HW/SW Co-Design Lab: Additional Information

Seminar 1

Slide 14:

TIE operation construct (1-cycle instruction):

```
operation operation_name {in/out regs} {in/out states/interfaces}
{
      operation_body
      wire, assign
}
```

TIE Regfile:

```
regfile reg128 128 8 rf
```

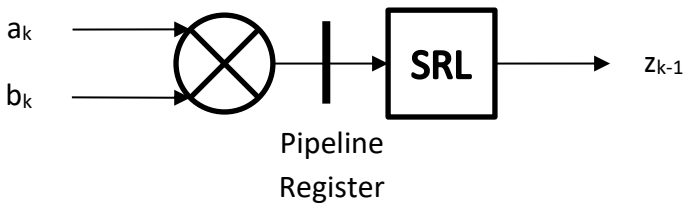
TIE States (instruction-specific regs):

```
state s128 128
```

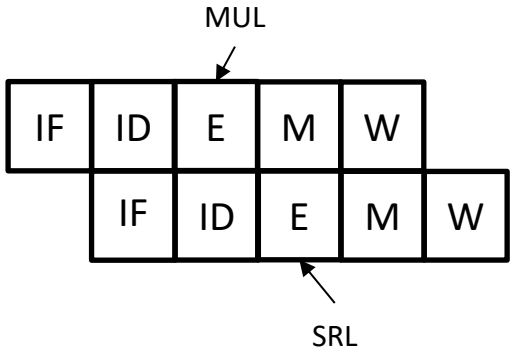
Slide 15:

Schedule operator:

```
schedule schedule_name {operation_name}
{
      schedule_body
      use, def
}
```



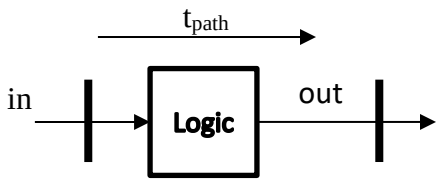
➔ After introducing pipeline register, output is delayed by 1



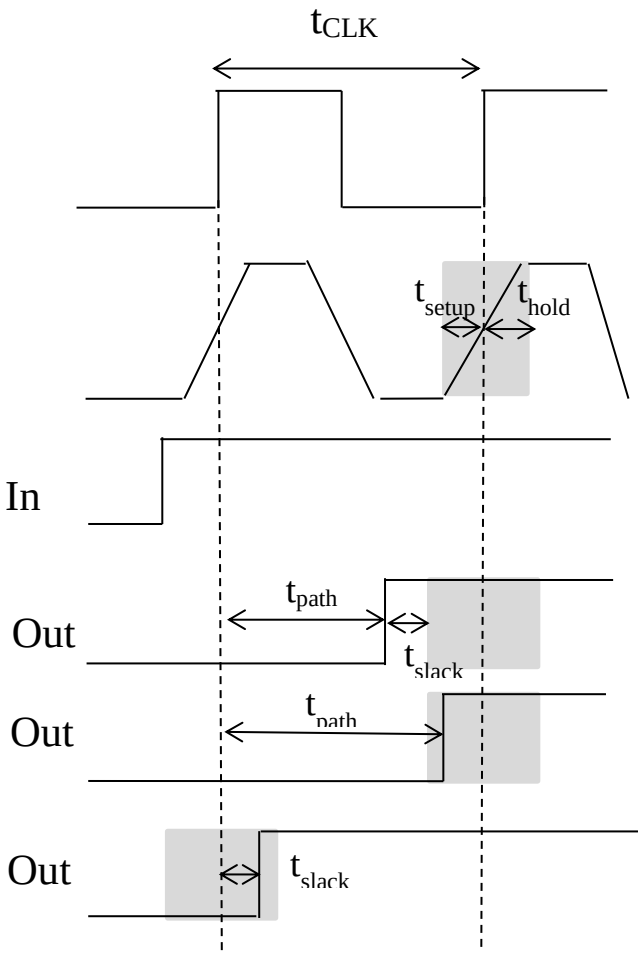
Why do we do pipelining? → Explain critical path

t_{MUL}, t_{ADD} → $t_{path} = t_{MUL} + t_{ADD} < t_{CLK} - t_{setup}$

$t_{slack} = t_{CLK} – t_{setup} – t_{path}$ → < 0, timing violated; > 0, timing met



- ➔ Setup and hold time requirements at input and out registers
- ➔ Setup violation can be avoided by reducing clock frequency
- ➔ Hold violation occurs due to very short-time combinational logic – can be avoided by introducing buffers



Timing met: $t_{slack} > 0$

Setup violation:
$t_{slack\text{-}setup} < 0$

Hold violation:
$t_{slack\text{-}hold} < 0$

Slide 35:

Approach 1)

$$y_n = \sum_{i=0}^{N-1} f(b_{i,} x_{k-i})$$

➔ Product of coefficients and inputs is a function with two inputs

TIE code:

```
operation FIR_MAC {inout AR y, in AR x, in AR b}{}
{
      wire [31:0] prod = TIEmul(x[15:0],b[15:0],1'b1);
      assign y = y + prod;
}
```

C code:

```
for(n = 0; n < len+7; n++)
      for(i = 0; i < 8; i++)
            FIR_MAC(out[n],in[n+7-i],coeff[i]);
```

Approach 2)

$$y_n = \begin{cases} b_0 x_n + \\ b_1 x_{n-1} + \\ b_2 x_{n-2} + \\ \cdots \\ b_7 x_{n-7} \end{cases}$$

➔ Take operation FIR_MAC with 128-bit regfile → 8x parallel multiplication

```
for(n = 0; n < len+7; n++)
      out[n] = FIR_MAC_SIMD(in[n+7],*coeff);
```

→ for all 8 coefficients

TIE code:

```
regfile fir_vec 128 8 fir_vec
```

Approach 3)

Instructions

1. Load x_{n-i} from memory
2. Read constants b_i from register
3. FIR_MAC(b_i, x_{n-i})
4. Store y_n to memory

TIE code:

```
state ptr_in 32
State data_in128 128

operation FIR_LD {}{out VAddr, in MemDataIn128, inout ptr_in,
                        out data_in128}
{
      assign VAddr = ptr_in;
      assign ptr_in = ptr_in + 16;
      assign data_in128 = MemDataIn128;
}
```

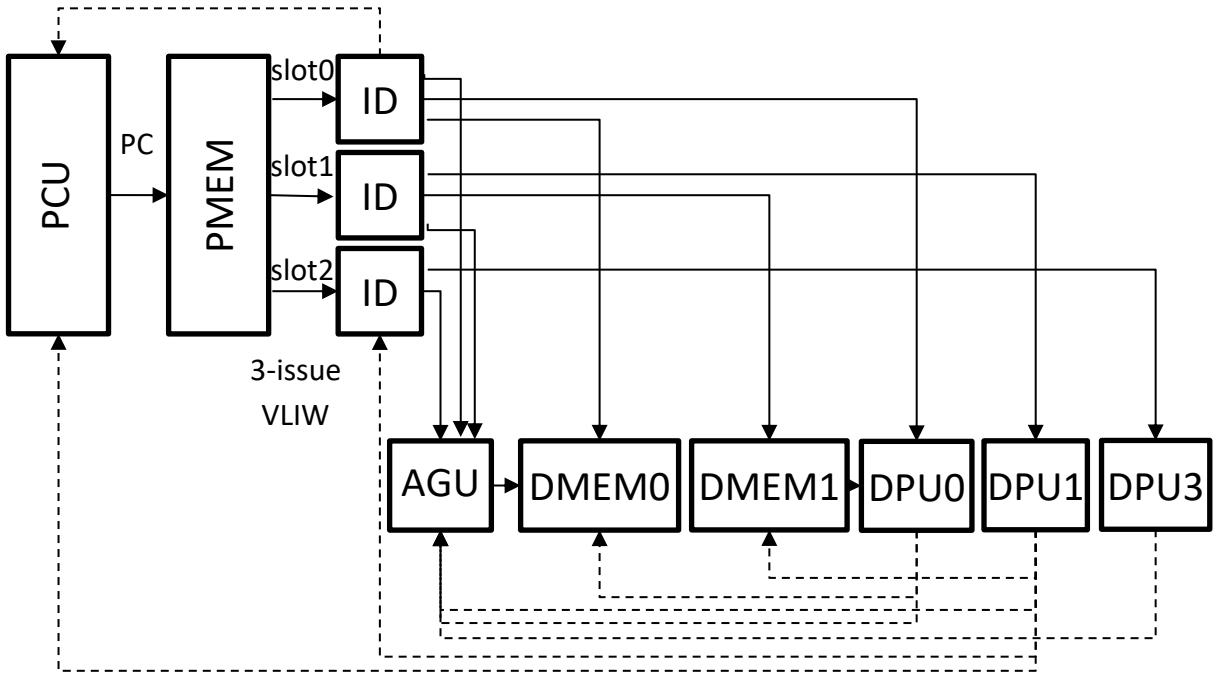
→ data_in128 is input (as state) for FIR_MAC_SIMD

Seminar 2

Slide 6:

VLIW Processor: important → 2x DMEM, 3x DPU

(do not draw dashed lines on board, just for completeness)



Slide 8:

3 Approaches:

1)

$$y_n = \sum_{i=0}^{N-1} op(b_i, x_{n-i})$$

2)

$$y_n = \sum_{\substack{i=0 \\ i=i+8}}^{N-1} \begin{cases} b_i x_{n-i} + \\ b_{i+1} x_{n-i-1} + \\ \dots \\ b_{i+7} x_{n-i-7} \end{cases} = \begin{cases} b_0 x_n + \\ b_1 x_{n-1} + \\ \dots \\ b_7 x_{n-7} \end{cases} + \dots$$

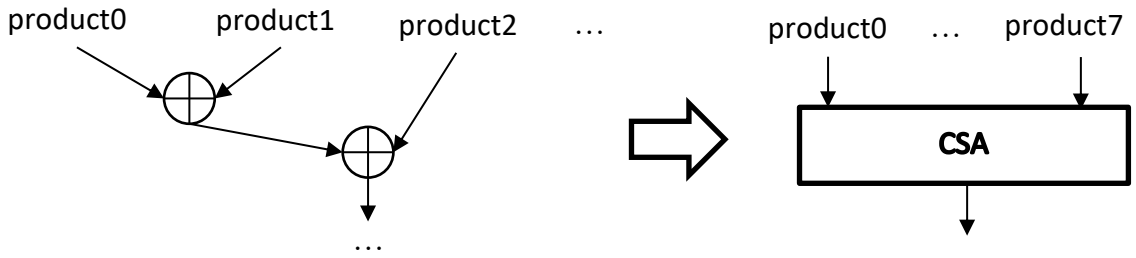
3)

(1) load x_{n-i}
(2) hold b_i in TIE table
(3) mac b, x
(4) store y_n

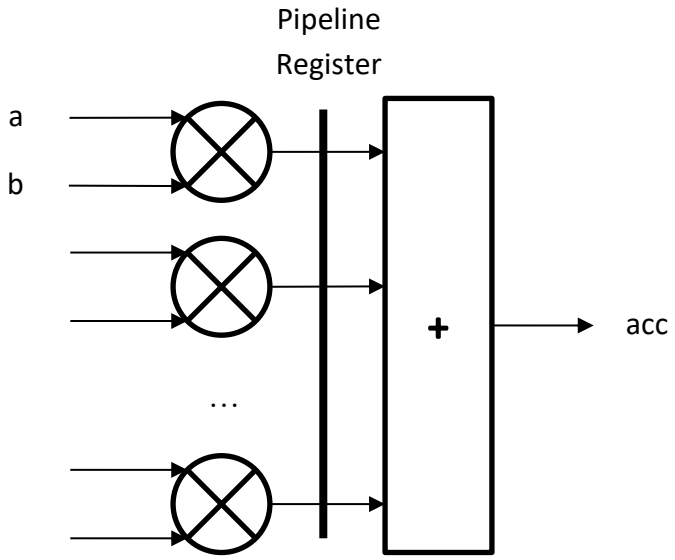
Slide 10:

By using TIEaddn() carry-save adder (CSA) instead of carry-ripple adder (CRA) is used

CRA:



Schedule leads to the following behavior



Corresponding C code (better show this directly in the demo in Xplorer):

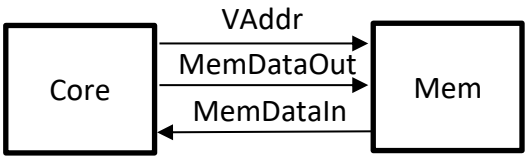
```
VR_fir *vr_p_coeff = (VR_fir*)coeff;
VR_fir vr_inpArr_slice;

out[n] = FIR_MAC_SIMD(vr_inpArr_slice, *vr_p_coeff);
```

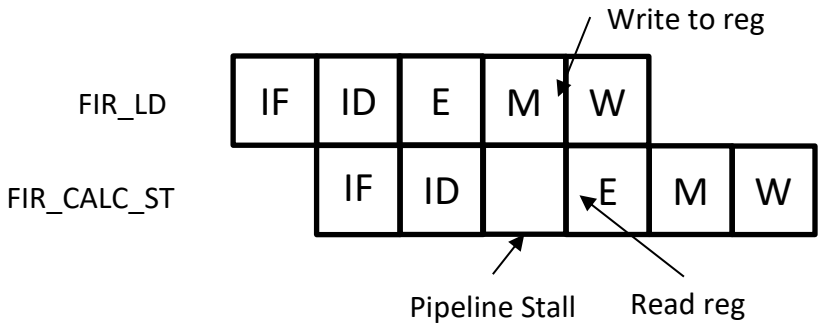
Slide 11:

Separate TIE operations: FIR_LD, FIR_CALC_ST
States: ptr_in, in128_2

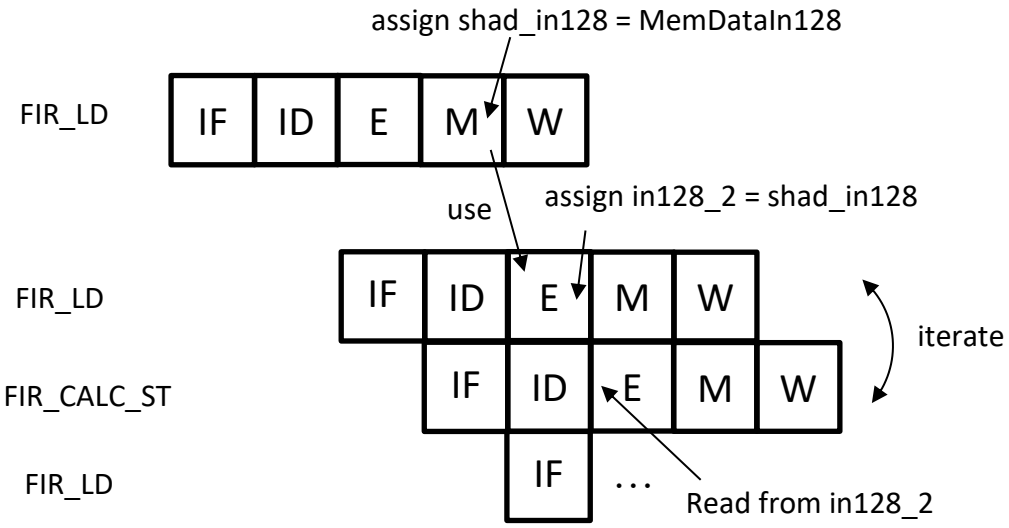
Using Memory Interface provided in TIE code: VAddr, MemDataInX, MemDataOutX



Without Shadow register:



With Shadow register:



Slide 14:

k: time index
m: frequency index
N: number of time/frequency values (power of 2)

Slide 15:

X_m with m = 0:

$$X_0 = \sum_{k=0}^{n-1} x_{even,k} + 1 \sum_{k=0}^{n-1} x_{odd,k} = \sum_{k=0}^{2n-1} x_n$$

With Twiddle factor:

$$W_n^m = e^{-j\pi\frac{m}{n}}$$
$$W_n^0 = e^{-j\pi\frac{0}{n}} = 1$$

Slide 16:

When looking at the Butterfly:

$$X_0 = \begin{Bmatrix} x_0 + W_1^0 x_4 \\ (x_2 + W_1^0 x_6) W_2^0 \end{Bmatrix} + \begin{Bmatrix} \begin{pmatrix} x_1 + W_1^0 x_5 \\ (x_3 + W_1^0 x_7) W_2^0 \end{pmatrix} W_4^0 \end{Bmatrix} + = \begin{Bmatrix} x_0 + x_4 \\ x_2 + x_6 \end{Bmatrix} + \begin{Bmatrix} x_1 + x_5 \\ x_3 + x_7 \end{Bmatrix} +$$

with: $W_1^0 = W_2^0 = W_4^0 = 1$

Slide 20:

DIF Butterfly