Frontier Design

http://www.frontierd.com

A|RT™ Designer

Instructive demo

Case Study

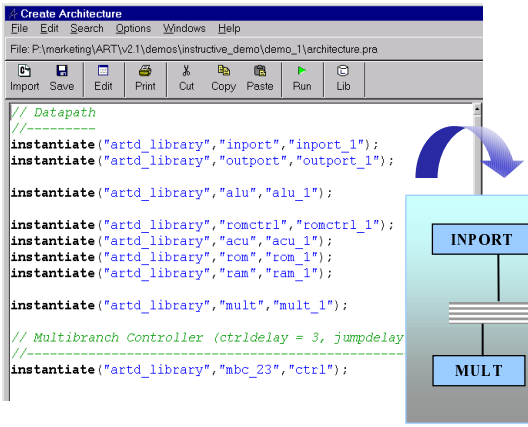
```
#include <fxp.h>

void demo (
   const Fix<8,7> A[8],
   const Fix<8,7> B[8],
   Fix<8,7>& overall_result,
   Uint<4>& position)
{
   #pragma OUT overall_result position

   static Fix<8,7> result = 0;
   Fix<8,7> current_result = 0;
   Uint<4> max= "0bu1111";

   for (Uint<4> i = 0; i < 8; i++)
   {
      Fix<8,7> product = A[i] * B[7-i];
      if (product > current_result)
      {
         current_result = -product;
         position = max+1;
      } // end if
      else
      {
         current_result = product<<2;
         position = max-1;
      } // end else
   }  // end for
   result = result + current_result;
   overall_result = result;
}
```

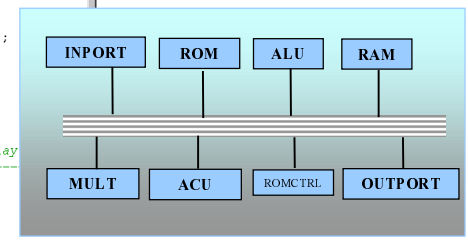
Algorithm

- Two arrays are multiplied pair-wise using a loop
- On the product some conditional operations are performed.

Case Study

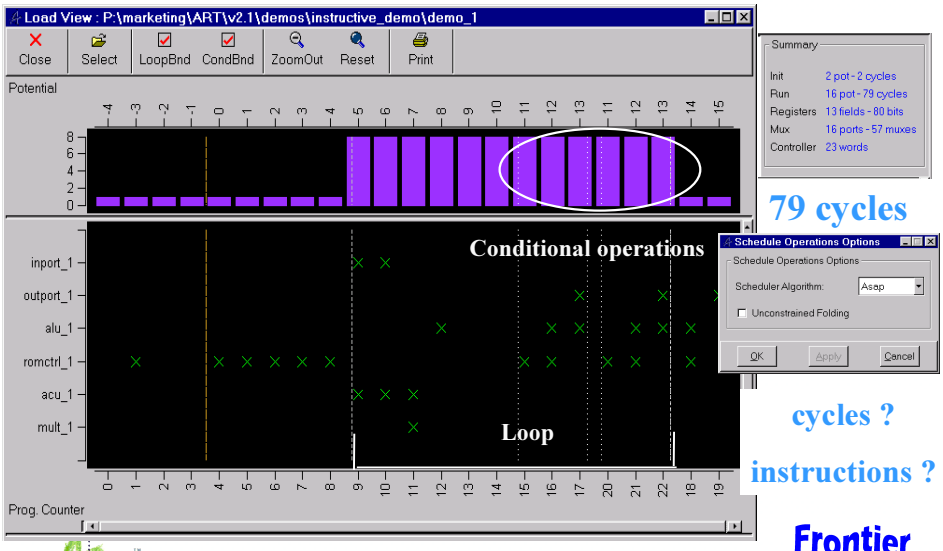
Initial architecture





Case Study

First run



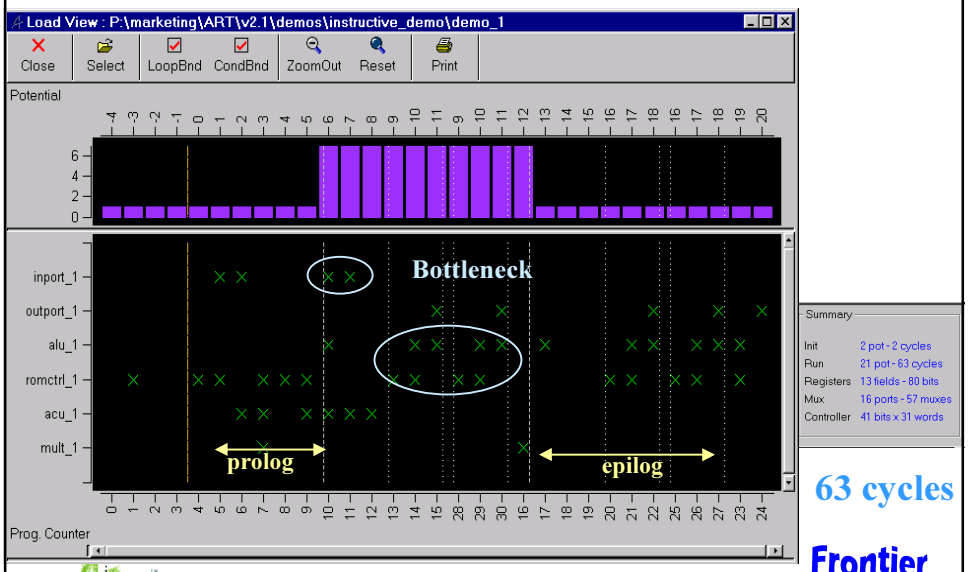
79 cycles

cycles ?

instructions ?

Case Study

Rerun with loopfolding



63 cycles

Case Study

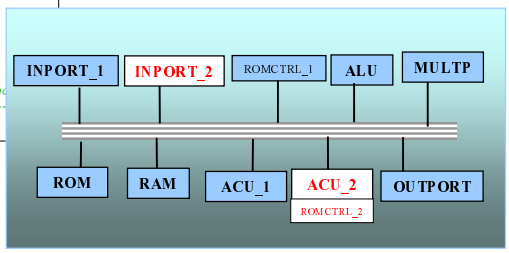
Further acceleration: extra resources

```
// Datapath
//---------
instantiate("artd_library","inport","inport_1");
instantiate("artd_library","inport","inport_2");
instantiate("artd_library","outport","outport_1");

instantiate("artd_library","alu","alu_1");

instantiate("artd_library","romctrl","romctrl_1");
instantiate("artd_library","romctrl","romctrl_2");
instantiate("artd_library","acu","acu_1");
instantiate("artd_library","acu","acu_2");
instantiate("artd_library","rom","rom_1");
instantiate("artd_library","ram","ram_1");

instantiate("artd_library","mult","mult_1");

// Multibranch Controller (ctrldelay = 3, jump
//-----------------------------------------------
instantiate("artd_library","mbc_23","ctrl");
```

```
assign_variable("/demo/B","inport_2");
dedicate("acu_2","inport_2");
dedicate("romctrl_2","acu_2");
```

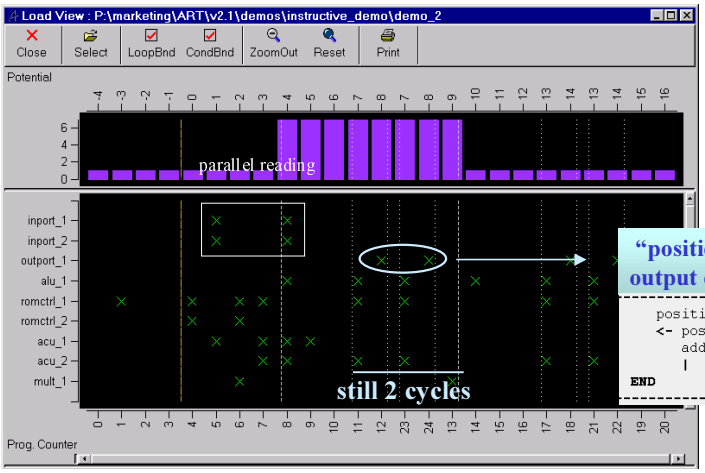
Use second acu to increase parallelism with alu

```
assign_operation("/demo/loopi/ifproduct/incr","acu_2");
assign_operation("/demo/loopi/ifproduct/decr","acu_2");
```



Case Study

Results



53 cycles

"position" is written to the output every! loop-iteration

```
position:outport_1
<- position$24:reg_d_outport_1,
   addr$34:reg_a_outport_1
   |  outport_1=write;
END
```


Case Study

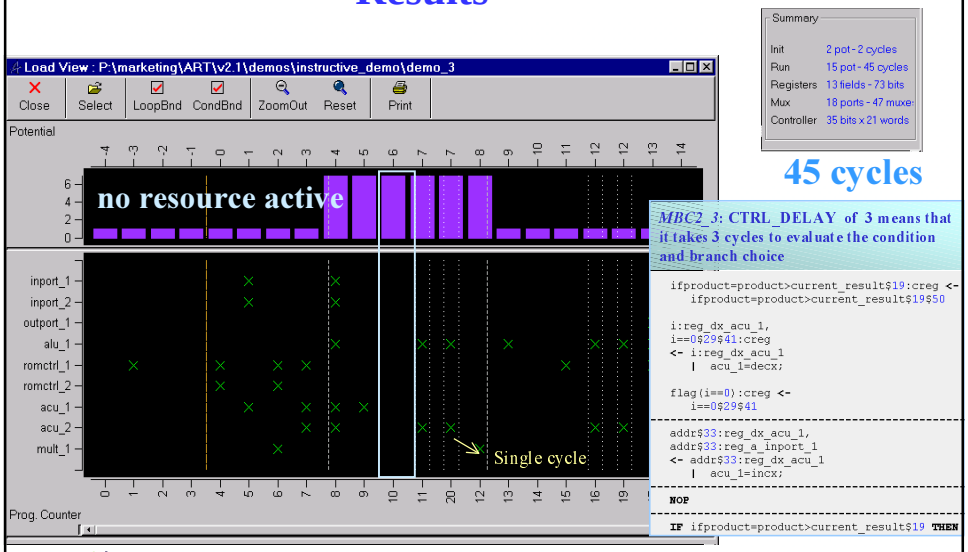
Solution: algorithm change

```
#include <fxp.h>

void demo (
   const Fix<8,7> A[8],
   const Fix<8,7> B[8],
   Fix<8,7>& overall_result,
   Uint<4>& position)
{
   #pragma OUT overall_result position

   static Fix<8,7> result = 0;
   Fix<8,7> current_result = 0;
   Uint<4> max = "0bu1111";
   Uint<4> tmp_pos;


   loopi: for (Uint<4> i = 0; i < 8; i++)
   {
      Fix<8,7> product = A[i] * B[7-i];
      ifproduct: if (product > current_result)
      {
         current_result = -product;
         incr: tmp_pos = max+1;
      } // end if
      else
      {
         current_result = product << 2;
         decr: tmp_pos = max-1;
      } // end else

   }  // end for
   result = result + current_result;
   overall_result = result;
   position = tmp_pos;
}
```

A "temporary" variable is introduced, so that only at the end of the function there is a write to the outport

Case Study

Results



Summary

Init	2 pot - 2 cycles
Run	15 pot - 45 cycles
Registers	13 fields - 73 bits
Mux	18 ports - 47 muxes
Controller	35 bits x 21 words

45 cycles

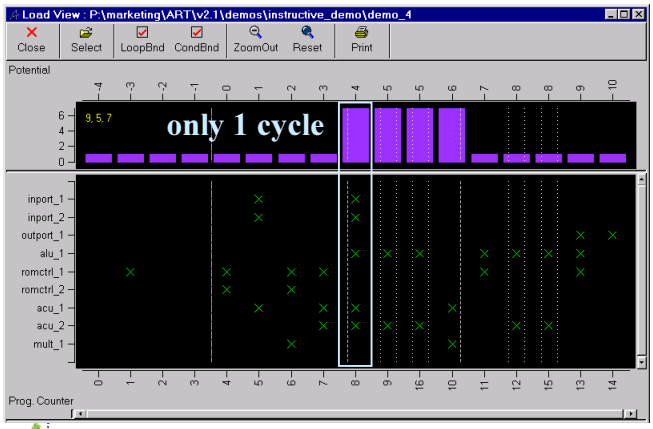
MBC2_3: CTRL_DELAY of 3 means that it takes 3 cycles to evaluate the condition and branch choice

```
ifproduct=product>current_result$19:creg <-
   ifproduct=product>current_result$19$50

i:reg_dx_acu_1,
i==0$29$41:creg
<- i:reg_dx_acu_1
   | acu_1=decx;

flag(i==0):creg <-
   i==0$29$41

addr$33:reg_dx_acu_1,
addr$33:reg_a_inport_1
<- addr$33:reg_dx_acu_1
   | acu_1=incx;

NOP

IF ifproduct=product>current_result$19 THEN
```


Case Study

alternative: mbc_11 controller

```
// Multibranch Controller (ctrldelay = 1, jumpdelay = 1)
//-----------------------------------------------------
instantiate("artd_library","mbc_11","ctrl");
```



Summary

Init	2 pot - 2 cycles
Run	11 pot - 29 cycles
Registers	13 fields - 73 bits
Mux	18 ports - 47 muxes
Controller	17 words

29 cycles

Case Study

Overview