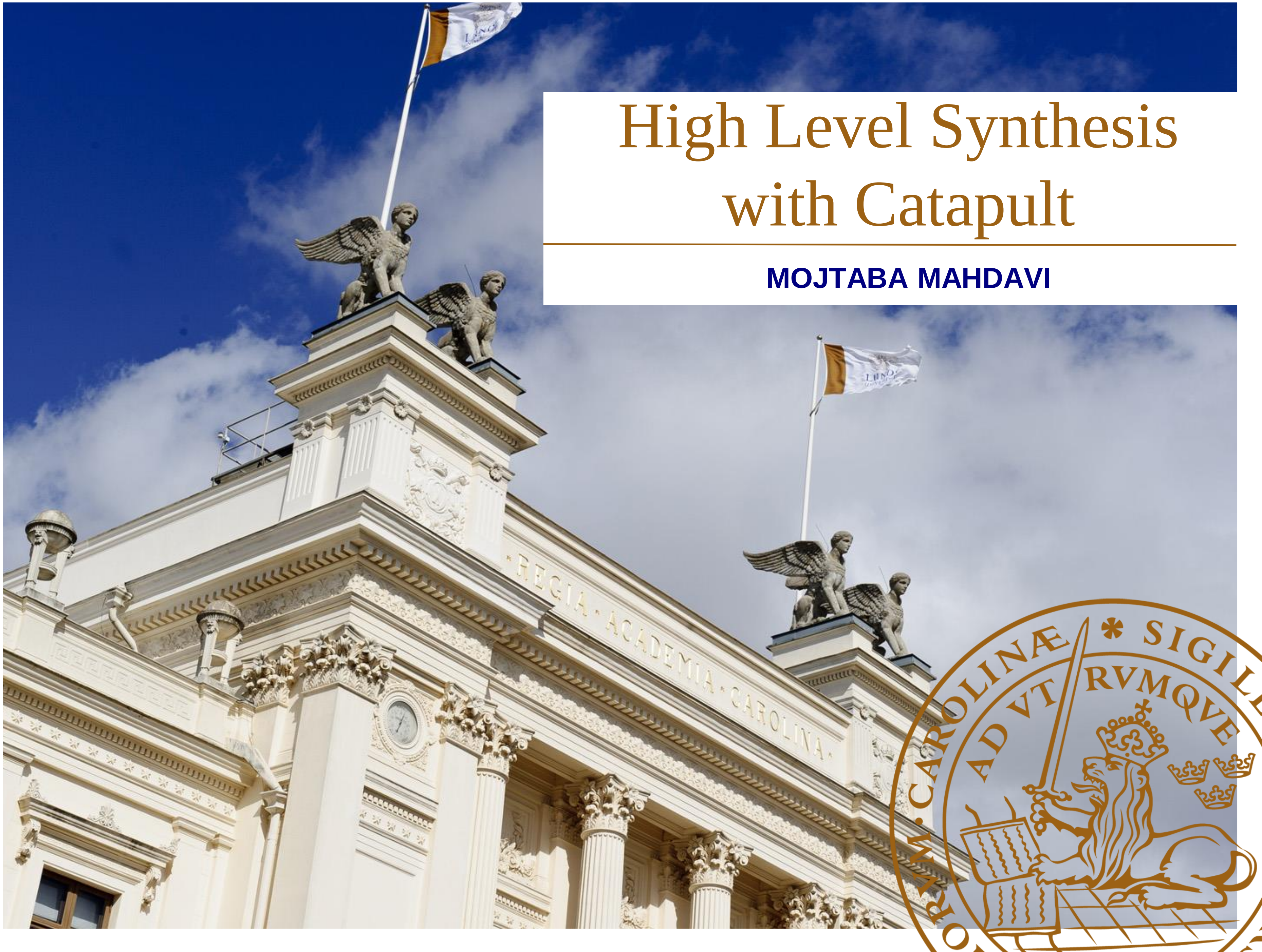


High Level Synthesis with Catapult

MOJTABA MAHDAVI

Outline

- ❑ High Level Synthesis (HLS)
- ❑ Catapult Basics
- ❑ Data Types
- ❑ Design Flow in Catapult

LUND UNIVERSITY

Outline

- ❑ High Level Synthesis (HLS)
- ❑ Catapult Basics
- ❑ Data Types
- ❑ Design Flow in Catapult

High Level Synthesis (HLS)

- ❑ Higher level of abstraction

- ❑ High-level synthesis bridges hardware and software domains.

- ❑ Generate hardware from C/C++ code or another high level language.

High Level Synthesis (HLS)

- A common misconception:
  - Synthesizing hardware from C++ provides users the freedom of describing the algorithms in any desired style of C++ coding.

- Remember that you are still describing hardware using a high level language !
  - "poor" description leads to a sub-optimal hardware.

High Level Synthesis (HLS)

- ❑It is important to know HW concepts to get optimal results

- ❑Successful projects require HW engineers not SW engineers
  - Hardware designers can work at a higher level of abstraction while creating high-performance hardware

High-Level Synthesis Benefits

- ❑ Develop algorithms at high level, e.g. C-level
  - Reduce design time
  - Reduce time for design changes
  - Easy design exploration
  - Faster time to market

- ❑ Verify at the higher level
  - Easier and faster verification

High-Level Synthesis Benefits

- Easy to explore different optimizations
  - Create multiple hardware implementation alternatives from the C source code using optimization directives

- Easy to maintain, easy to reuse
  - Create readable and portable C source code

Some Limitations

- Not all algorithms are suited for hardware implementations using HLS
  - Sequential code
  - Control logic
  - Large loops with function calls

Outline

- ❑ High Level Synthesis (HLS)
- ❑ Catapult Basics
- ❑ Data Types
- ❑ Design Flow in Catapult

Catapult

- *Catapult* is a product of Mentor Graphics, which is used for high-level synthesis.

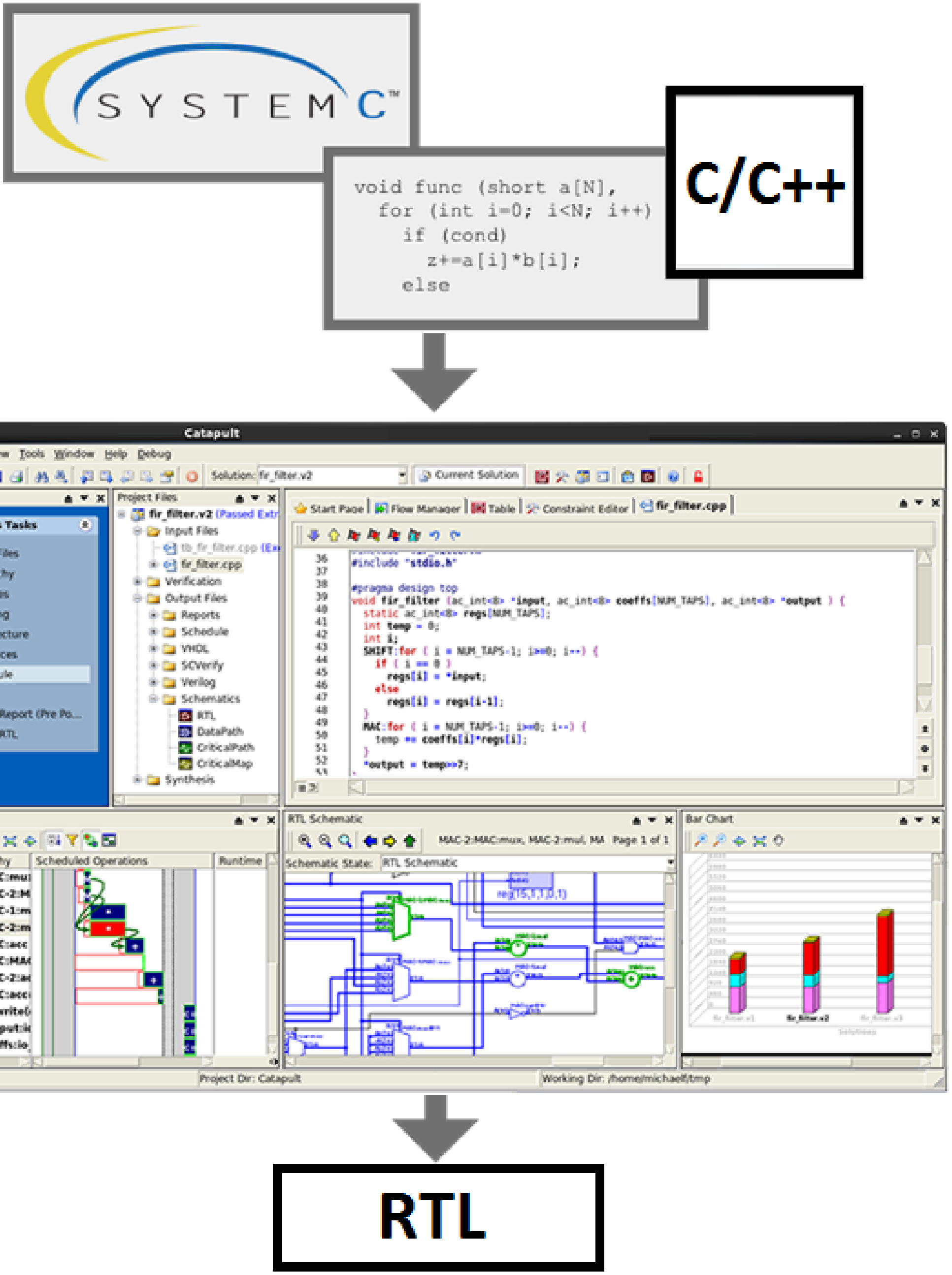
- Catapult takes C/C++ and System C as inputs and generates register transfer level (RTL) code.

- The target device can be FPGA and ASIC.

Catapult

- ❑The concepts that you have learned in the “DSP Design” course are used in the Catapult SW
  - Folding/Unfolding
  - Re-timing/Pipelining
  - Bit-level optimization
  - and more…

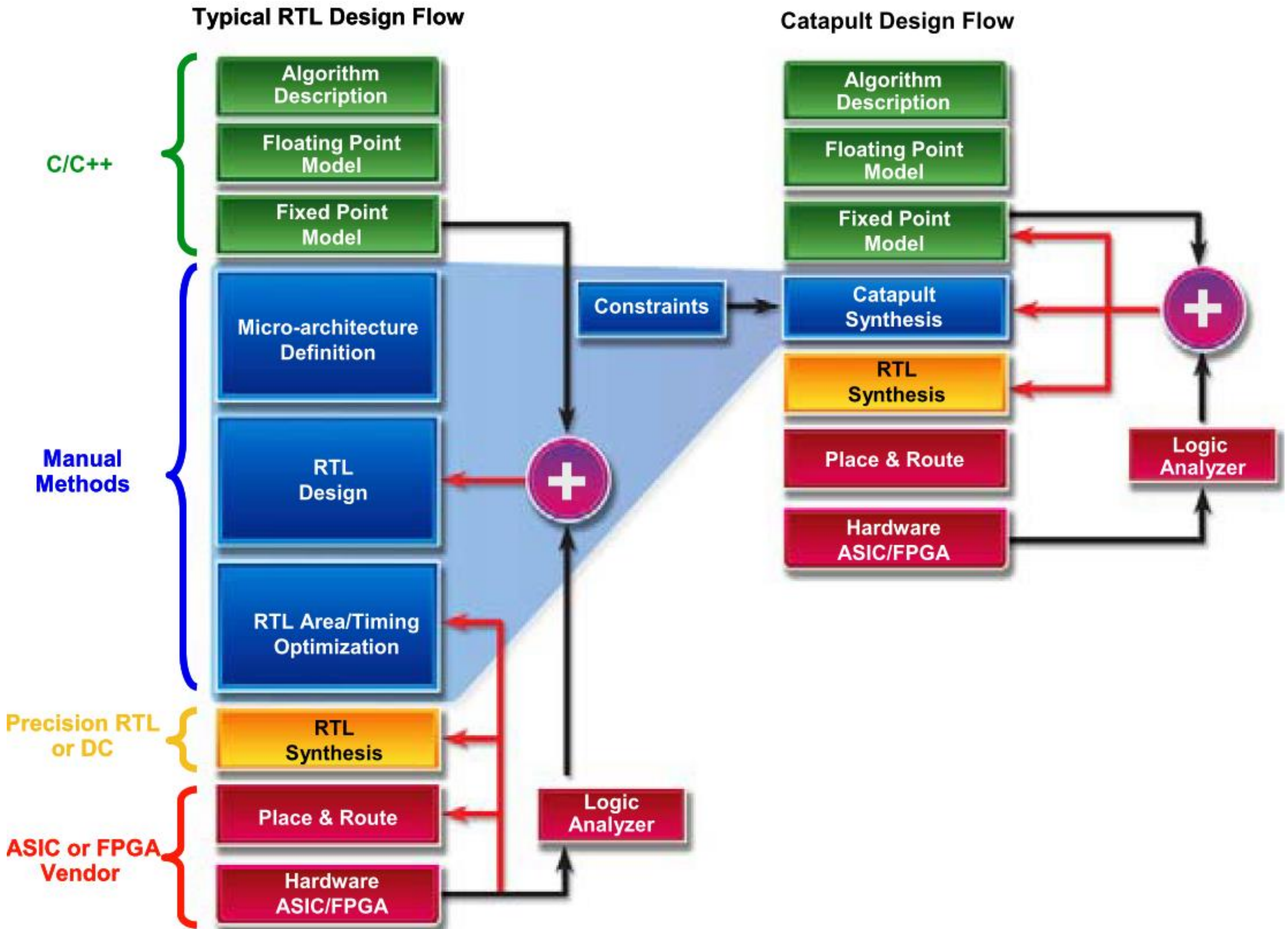
Catapult



```
void func (short a[N],
  for (int i=0; i<N; i++)
    if (cond)
      z+=a[i]*b[i];
    else
```

```
#include "stdio.h"

#pragma design top
void fir_filter (ac_int<8> *input, ac_int<8> coeffs[NUM_TAPS], ac_int<8> *output ) {
  static ac_int<8> regs[NUM_TAPS];
  int temp = 0;
  int i;
  SHIFT:for ( i = NUM_TAPS-1; i>=0; i--) {
    if ( i == 0 )
      regs[i] = *input;
    else
      regs[i] = regs[i-1];
  }
  MAC:for ( i = NUM_TAPS-1; i>=0; i--) {
    temp += coeffs[i]*regs[i];
  }
  *output = temp>>7;
```

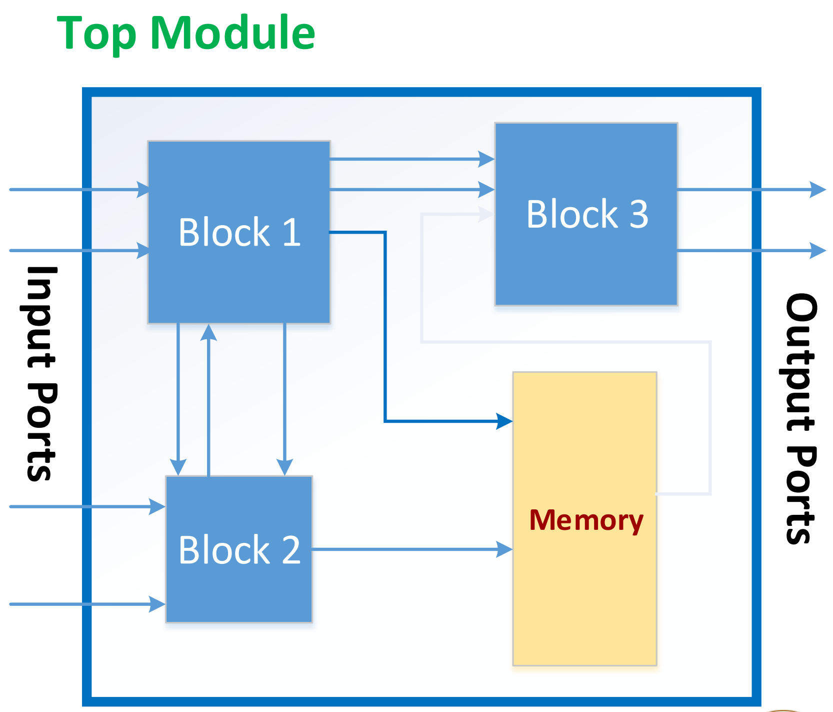
Traditional Design Flow vs. Catapult Flow



Top-level Design

- ❑ Similar to RTL designs, the "top" level of design should be specified in HLS design flow.
  - ○ The highest level of the design

- ❑ Top module specifies:
  - ○ The design interfaces with the outside world
  - ○ Port definitions, direction, bit widths, and data types



LUND UNIVERSITY

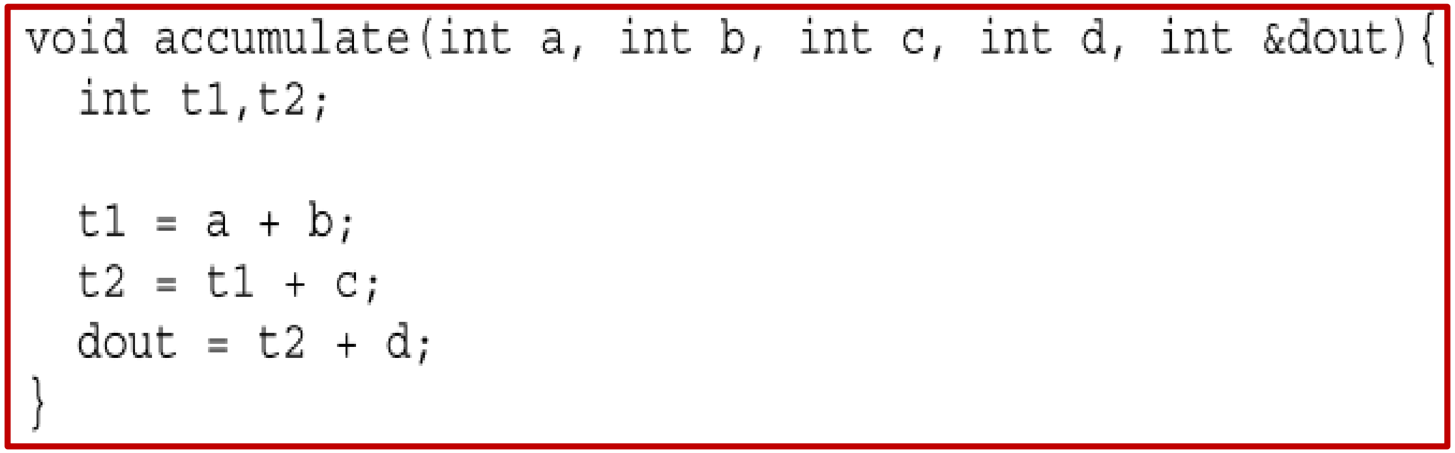
Registered Outputs

- In general, high-level synthesis builds synchronous designs by default.

- All outputs of the top-level design are registered
  - To guarantee that timing is met when connecting to another design

Design Example

- Consider the following example to introduce:
    - **HLS** concept
    - corresponding steps in **Catapult** design flow



```
void accumulate(int a, int b, int c, int d, int &dout){
  int t1,t2;

  t1 = a + b;
  t2 = t1 + c;
  dout = t2 + d;
}
```

Port Width and Direction

- ❑ Bit widths of the top-level ports, excluding clock and reset, are implied by the data type.
- ❑ The port direction is implied by how an interface variable is used in the C++ code.

Control Ports

- ❑ C++ code has no concept of timing.
- ❑ No clock, enable, or reset in the design source files.
  - o These signals are added by the synthesis tool.
- ❑ Control over things like polarity, type of reset, etc., are taken care by setting design constraints.

Outline

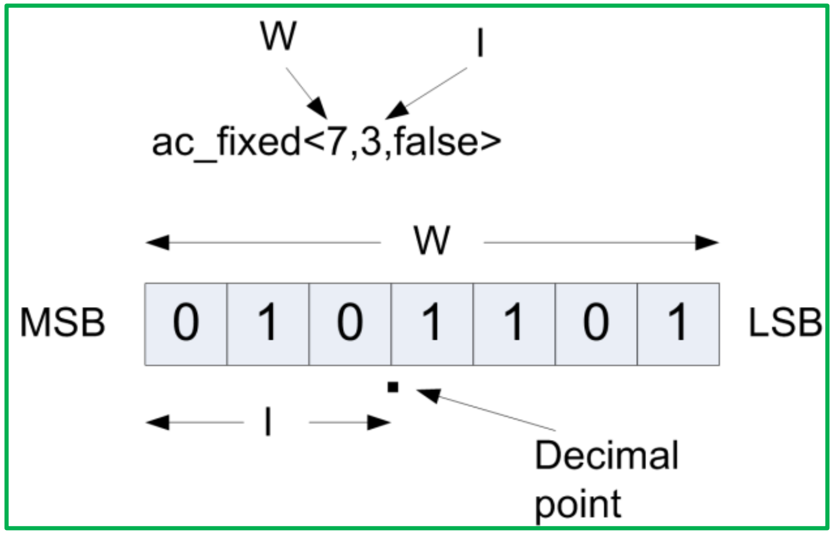
- ❑ High Level Synthesis (HLS)
- ❑ Catapult Basics
- ❑ Data Types
- ❑ Design Flow in Catapult

Data Types

- Two important data types:
  - int, ac_int
  - ac_fixed
- To use these data types the following header files should be included in the design source files:
  - “ac_int.h”
  - “ac_fixed.h”
- It is possible to define user data types.

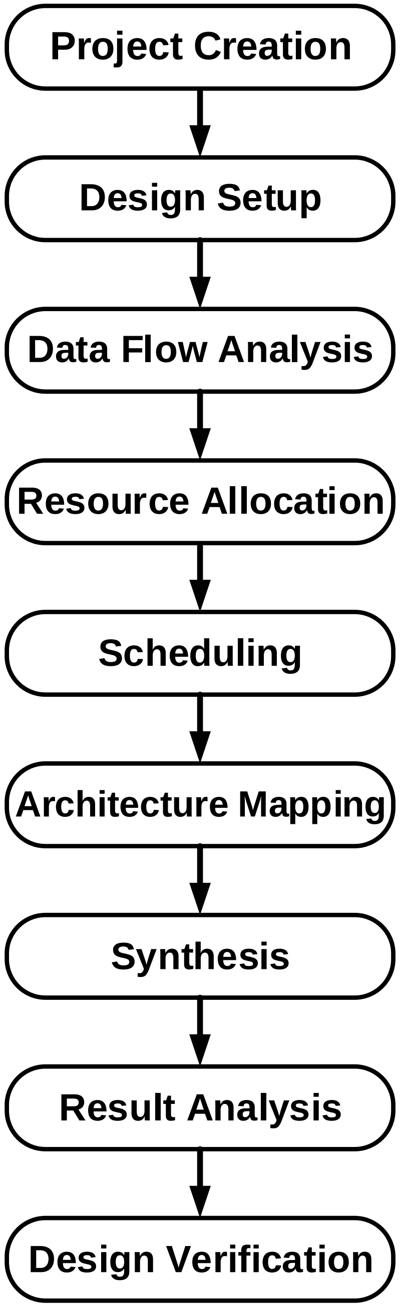
Data Types

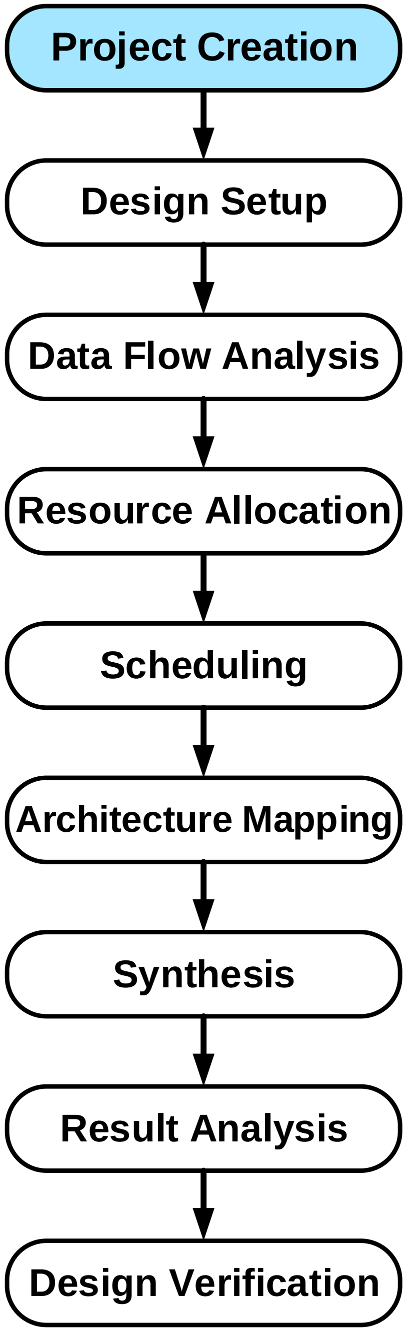
Type	Description	Numerical Range	Quantum
ac_int<W, false>	unsigned integer	0 to $2^W - 1$	1
ac_int<W, true>	signed integer	-2^{W-1} to $2^{W-1} - 1$	1
ac_fixed<W, I, false>	unsigned fixed-point	0 to $(1 - 2^{-W})\, 2^I$	2^{I-W}
ac_fixed<W, I, true>	signed fixed-point	$(-0.5)\, 2^I$ to $(0.5 - 2^{-W})\, 2^I$	2^{I-W}



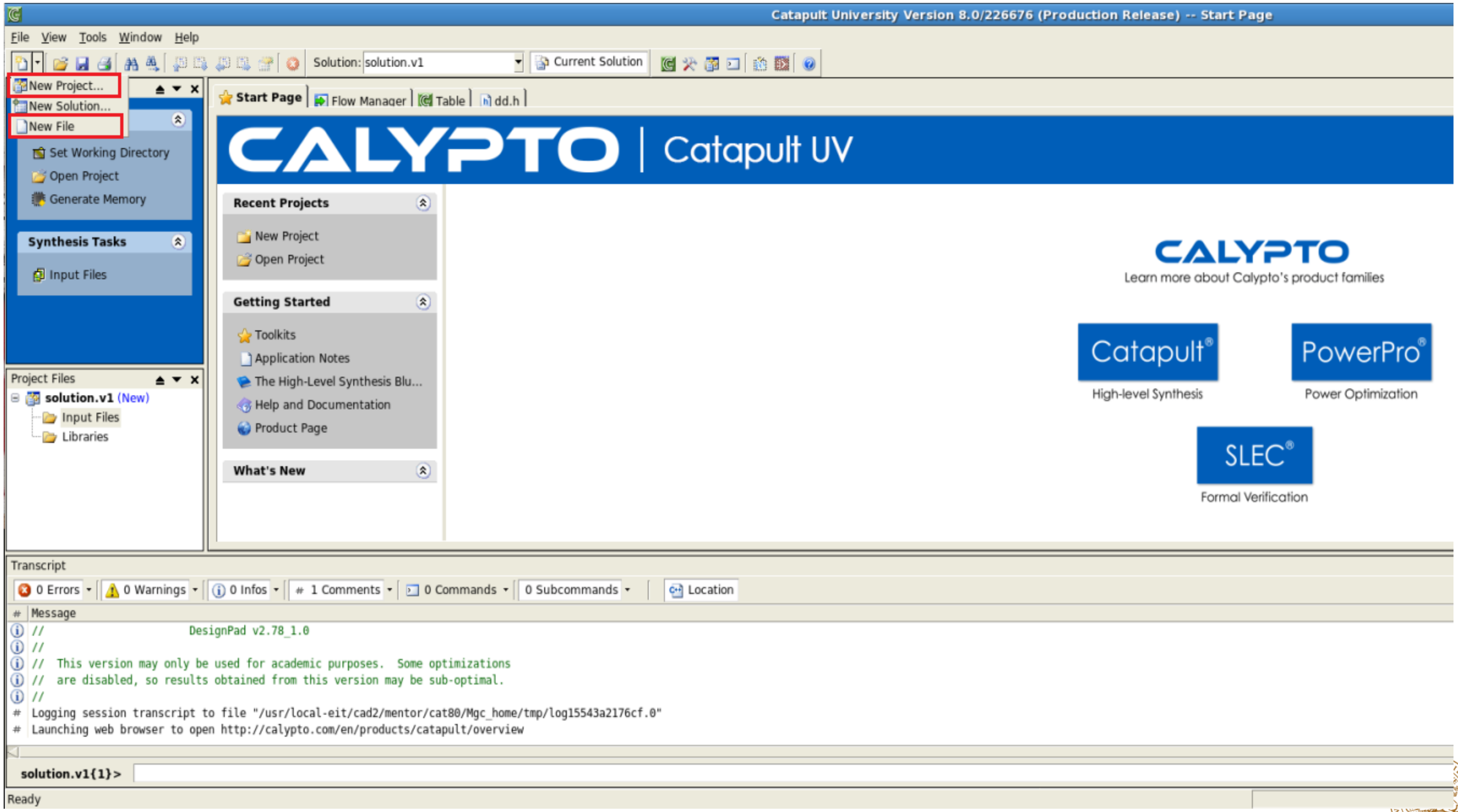
Outline

- ❑ High Level Synthesis (HLS)
- ❑ Catapult Basics
- ❑ Data Types
- ❑ Design Flow in Catapult


Project Creation
Design Setup
Data Flow Analysis
Resource Allocation
Scheduling
Architecture Mapping
Synthesis
Result Analysis
Design Verification

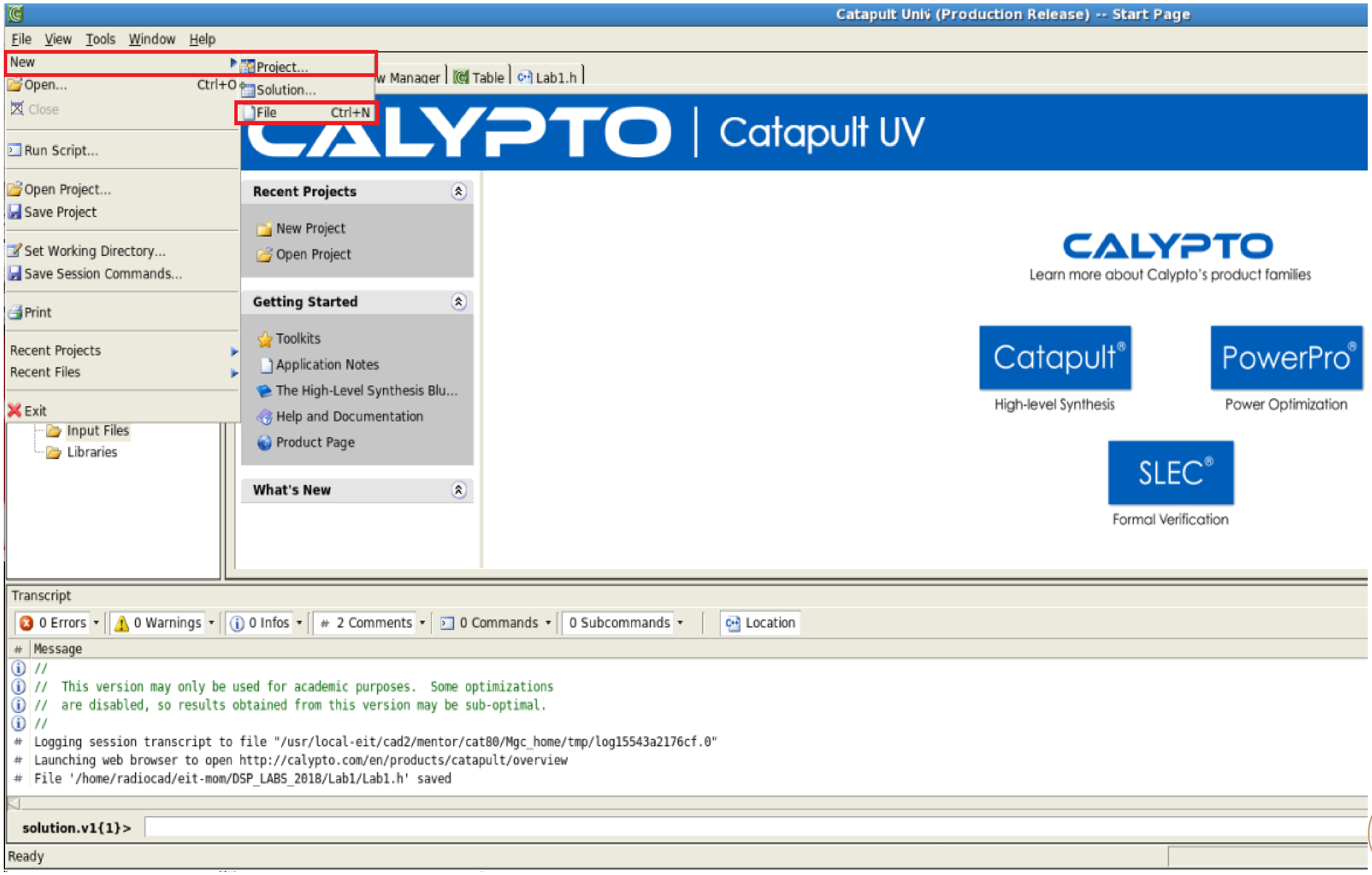

Project Creation
Design Setup
Data Flow Analysis
Resource Allocation
Scheduling
Architecture Mapping
Synthesis
Result Analysis
Design Verification

Create a New Project



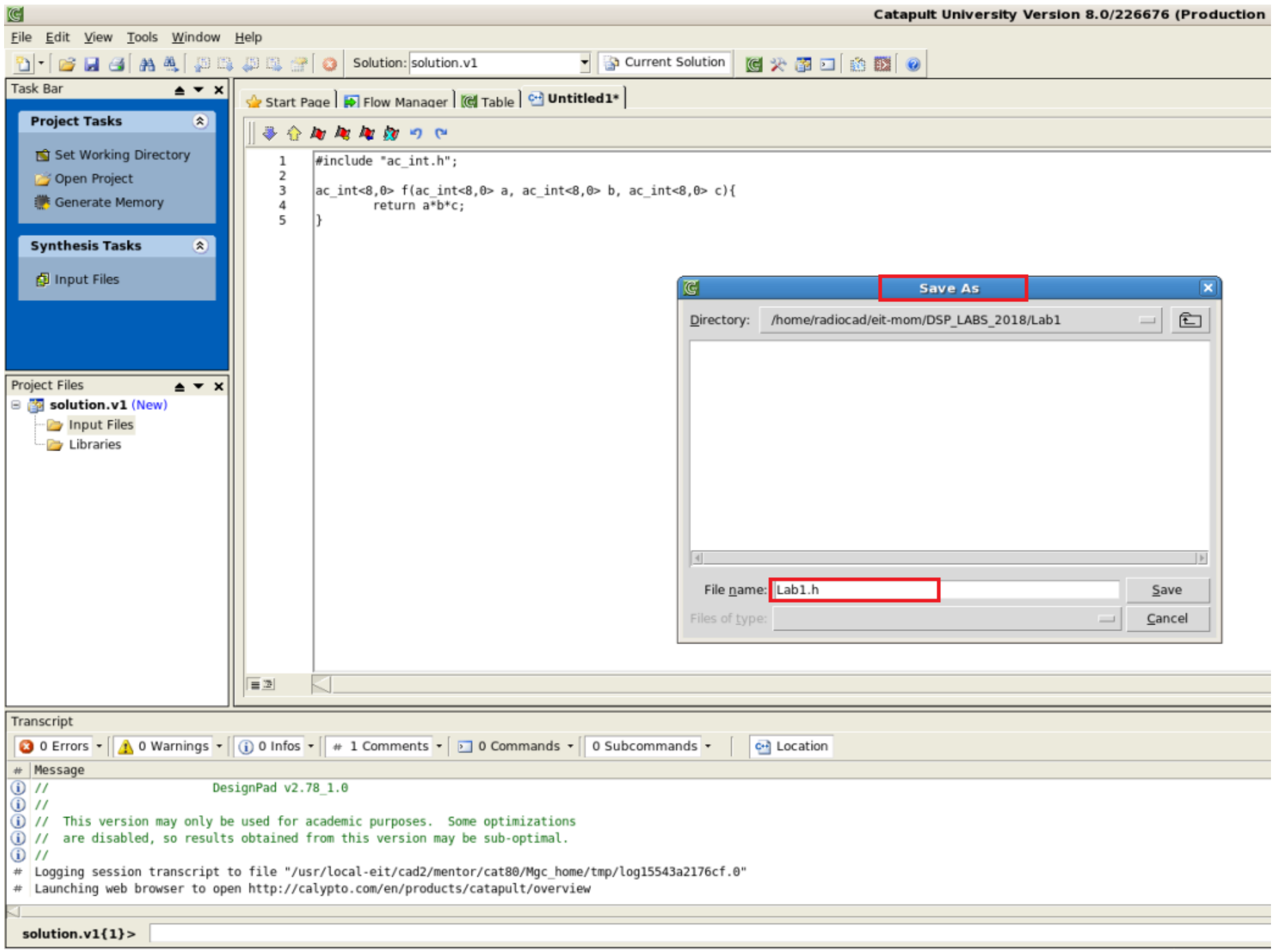
```
#  Message
//                 DesignPad v2.78_1.0
//
//  This version may only be used for academic purposes.  Some optimizations
//  are disabled, so results obtained from this version may be sub-optimal.
//
#  Logging session transcript to file "/usr/local-eit/cad2/mentor/cat80/Mgc_home/tmp/log15543a2176cf.0"
#  Launching web browser to open http://calypto.com/en/products/catapult/overview
solution.v1{1}>
```

Create a New Project



```
# Message
//
//  This version may only be used for academic purposes.  Some optimizations
//  are disabled, so results obtained from this version may be sub-optimal.
//
Logging session transcript to file "/usr/local-eit/cad2/mentor/cat80/Mgc_home/tmp/log15543a2176cf.0"
Launching web browser to open http://calypto.com/en/products/catapult/overview
File '/home/radiocad/eit-mom/DSP_LABS_2018/Lab1/Lab1.h' saved

solution.v1{1}>
```

Create a New Design File



```
#include "ac_int.h"

ac_int<8,0> f(ac_int<8,0> a, ac_int<8,0> b, ac_int<8,0> c){
        return a*b*c;
}
```

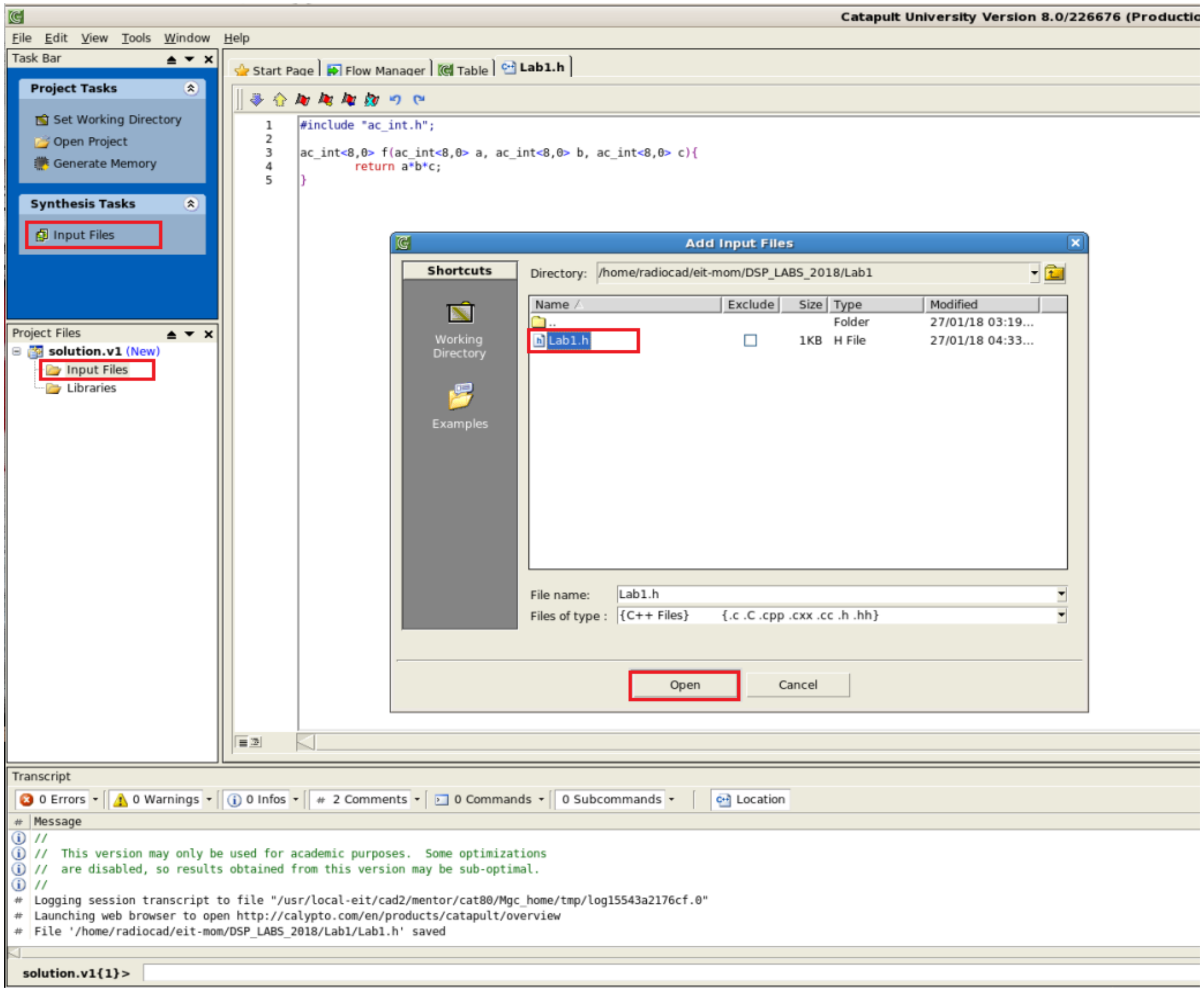
Directory: /home/radiocad/eit-mom/DSP_LABS_2018/Lab1

File name: Lab1.h

Files of type:

```
//                    DesignPad v2.78_1.0
//
//  This version may only be used for academic purposes.  Some optimizations
//  are disabled, so results obtained from this version may be sub-optimal.
//
Logging session transcript to file "/usr/local-eit/cad2/mentor/cat80/Mgc_home/tmp/log15543a2176cf.0"
Launching web browser to open http://calypto.com/en/products/catapult/overview
solution.v1{1}>
```

Add Input Files



```
#include "ac_int.h";

ac_int<8,0> f(ac_int<8,0> a, ac_int<8,0> b, ac_int<8,0> c){
        return a*b*c;
}
```

Directory: /home/radiocad/eit-mom/DSP_LABS_2018/Lab1

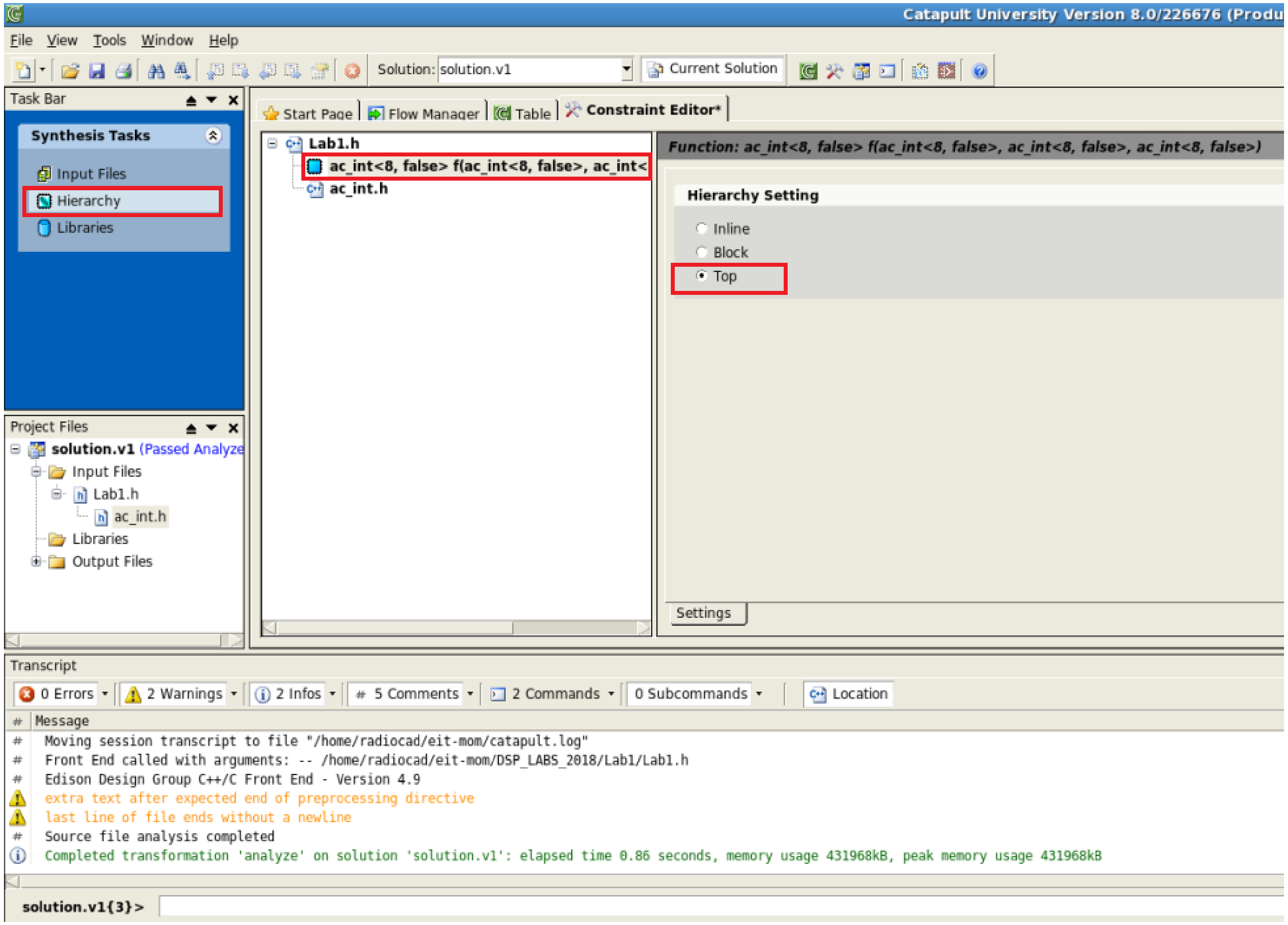
Name	Exclude	Size	Type	Modified
..			Folder	27/01/18 03:19...
Lab1.h	☐	1KB	H File	27/01/18 04:33...

File name: Lab1.h

Files of type : {C++ Files} {.c .C .cpp .cxx .cc .h .hh}

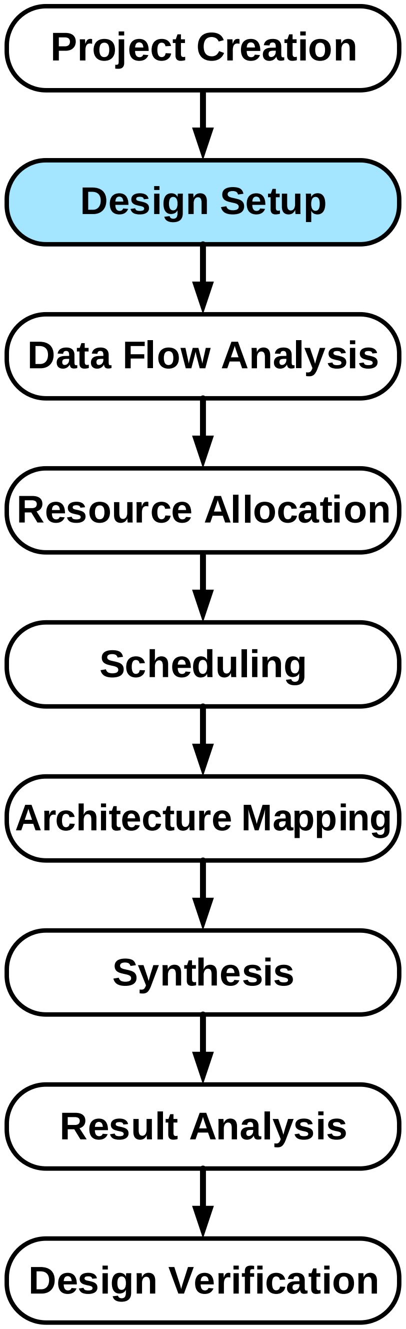
```
//
//  This version may only be used for academic purposes.  Some optimizations
//  are disabled, so results obtained from this version may be sub-optimal.
//
Logging session transcript to file "/usr/local-eit/cad2/mentor/cat80/Mgc_home/tmp/log15543a2176cf.0"
Launching web browser to open http://calypto.com/en/products/catapult/overview
File '/home/radiocad/eit-mom/DSP_LABS_2018/Lab1/Lab1.h' saved

solution.v1{1}>
```

Set Top Module

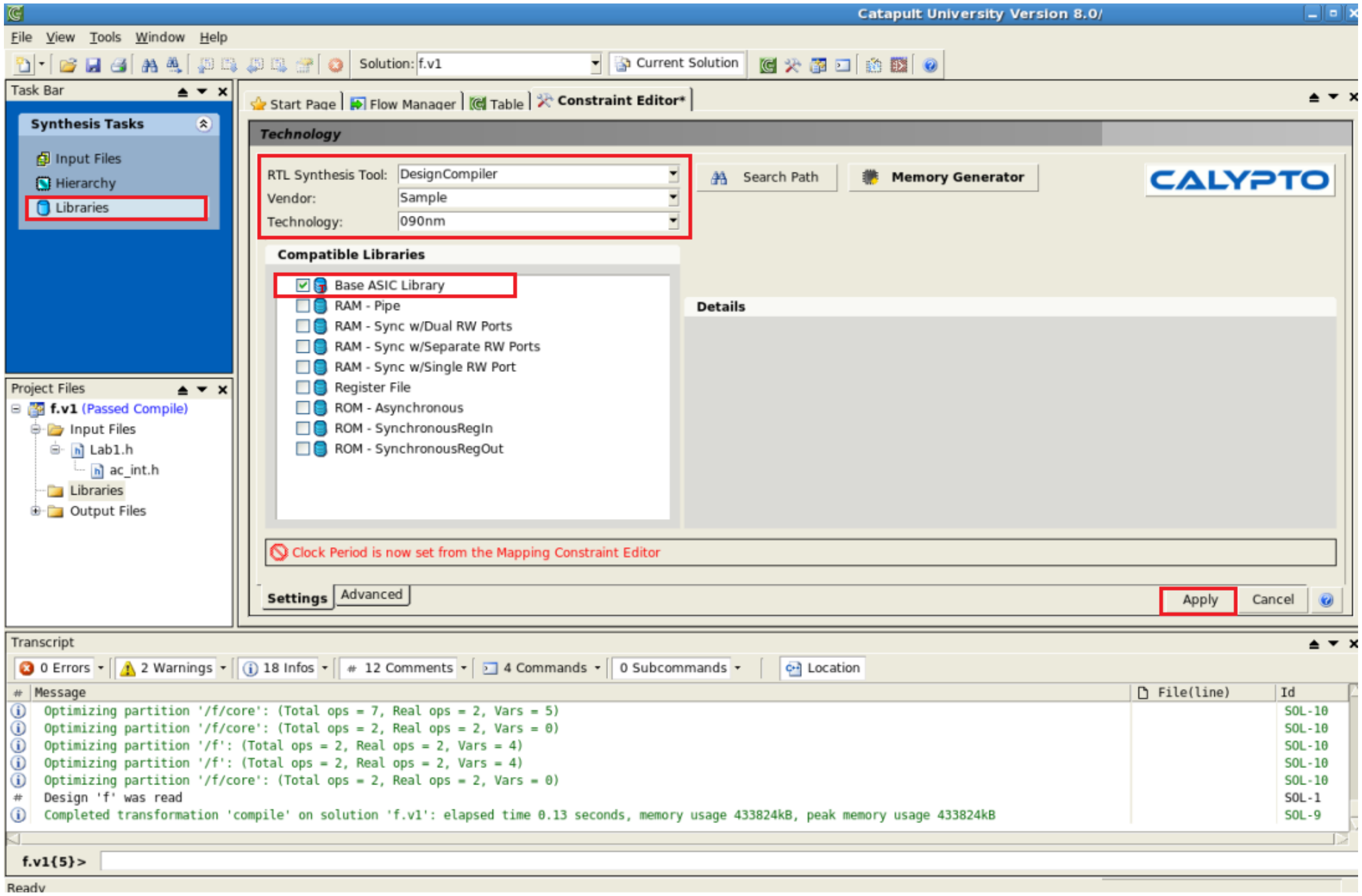


Hierarchy Setting: Inline [✓ Top] Block

```
#  Message
#  Moving session transcript to file "/home/radiocad/eit-mom/catapult.log"
#  Front End called with arguments: -- /home/radiocad/eit-mom/DSP_LABS_2018/Lab1/Lab1.h
#  Edison Design Group C++/C Front End - Version 4.9
   extra text after expected end of preprocessing directive
   last line of file ends without a newline
#  Source file analysis completed
   Completed transformation 'analyze' on solution 'solution.v1': elapsed time 0.86 seconds, memory usage 431968kB, peak memory usage 431968kB
solution.v1{3}>
```


Project Creation
Design Setup
Data Flow Analysis
Resource Allocation
Scheduling
Architecture Mapping
Synthesis
Result Analysis
Design Verification

Design Setup



RTL Synthesis Tool: DesignCompiler
Vendor: Sample
Technology: 090nm

Compatible Libraries:
- [✓] Base ASIC Library
- [ ] RAM - Pipe
- [ ] RAM - Sync w/Dual RW Ports
- [ ] RAM - Sync w/Separate RW Ports
- [ ] RAM - Sync w/Single RW Port
- [ ] Register File
- [ ] ROM - Asynchronous
- [ ] ROM - SynchronousRegIn
- [ ] ROM - SynchronousRegOut

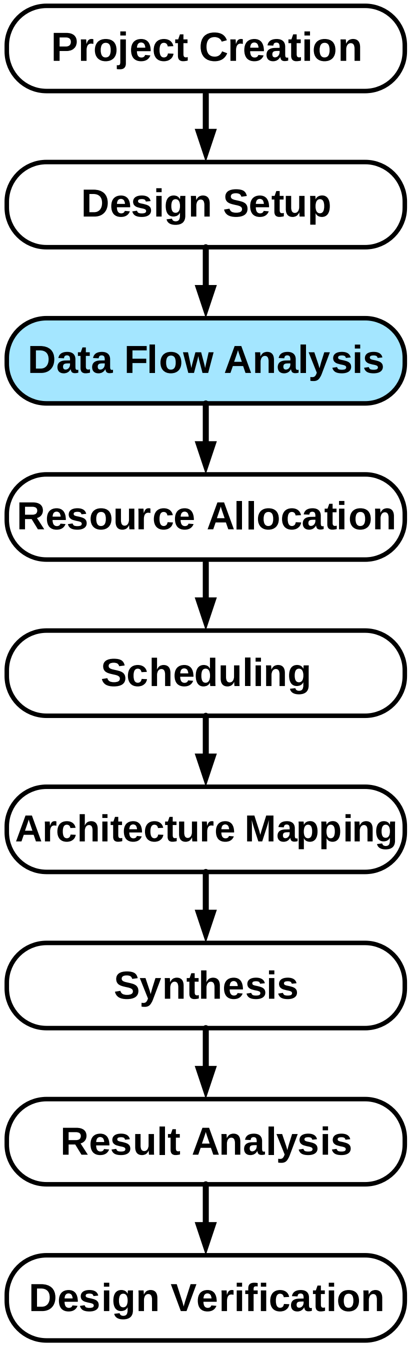
Clock Period is now set from the Mapping Constraint Editor

Project Files:
- f.v1 (Passed Compile)
  - Input Files
    - Lab1.h
      - ac_int.h
  - Libraries
  - Output Files

0 Errors | 2 Warnings | 18 Infos | 12 Comments | 4 Commands | 0 Subcommands

```
#  Message                                                                                                  File(line)  Id
   Optimizing partition '/f/core': (Total ops = 7, Real ops = 2, Vars = 5)                                              SOL-10
   Optimizing partition '/f/core': (Total ops = 2, Real ops = 2, Vars = 0)                                              SOL-10
   Optimizing partition '/f': (Total ops = 2, Real ops = 2, Vars = 4)                                                   SOL-10
   Optimizing partition '/f': (Total ops = 2, Real ops = 2, Vars = 4)                                                   SOL-10
   Optimizing partition '/f/core': (Total ops = 2, Real ops = 2, Vars = 0)                                              SOL-10
#  Design 'f' was read                                                                                                  SOL-1
   Completed transformation 'compile' on solution 'f.v1': elapsed time 0.13 seconds, memory usage 433824kB, peak memory usage 433824kB   SOL-9

f.v1{5}>
```

LUND UNIVERSITY


Project Creation
Design Setup
Data Flow Analysis
Resource Allocation
Scheduling
Architecture Mapping
Synthesis
Result Analysis
Design Verification

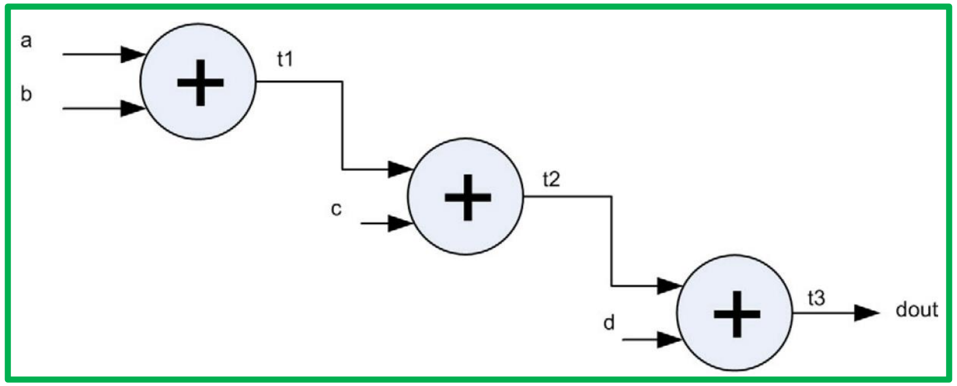
Data Flow Graph Analysis

- HLS process starts by analyzing the data dependencies between the various steps in the algorithm.

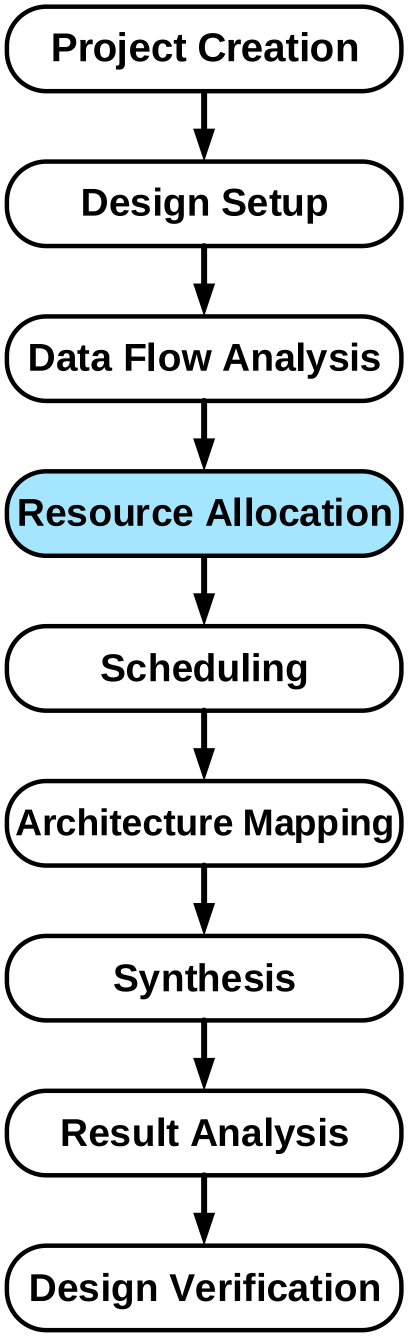
- The analysis leads to a Data Flow Graph (DFG).

Design Example

```
void accumulate(int a, int b, int c, int d, int &dout){
  int t1,t2;

  t1 = a + b;
  t2 = t1 + c;
  dout = t2 + d;
}
```



❑ Data dependencies and the order of operations are specified by the connection between nodes.


Project Creation
Design Setup
Data Flow Analysis
Resource Allocation
Scheduling
Architecture Mapping
Synthesis
Result Analysis
Design Verification

Resource Allocation

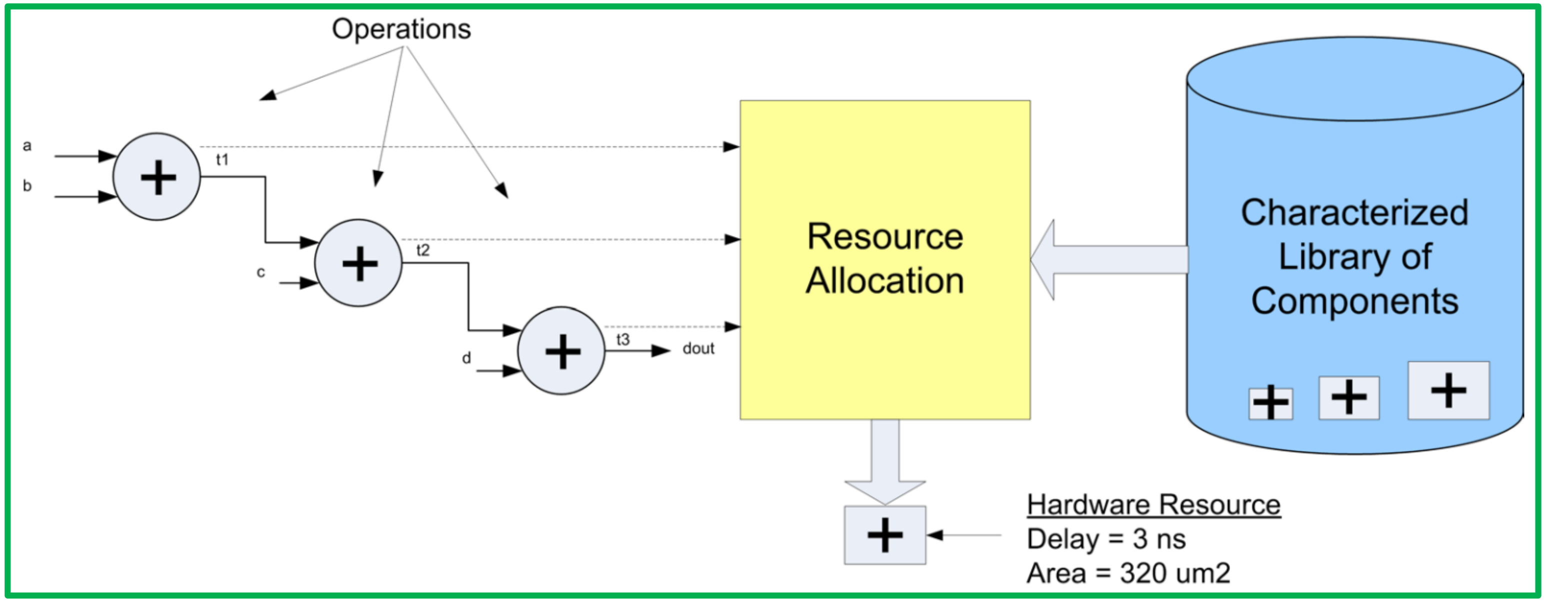
- Each operation in the extracted DFG is mapped onto a hardware resource.
  - Each resource corresponds to a physical implementation of the operator in hardware.

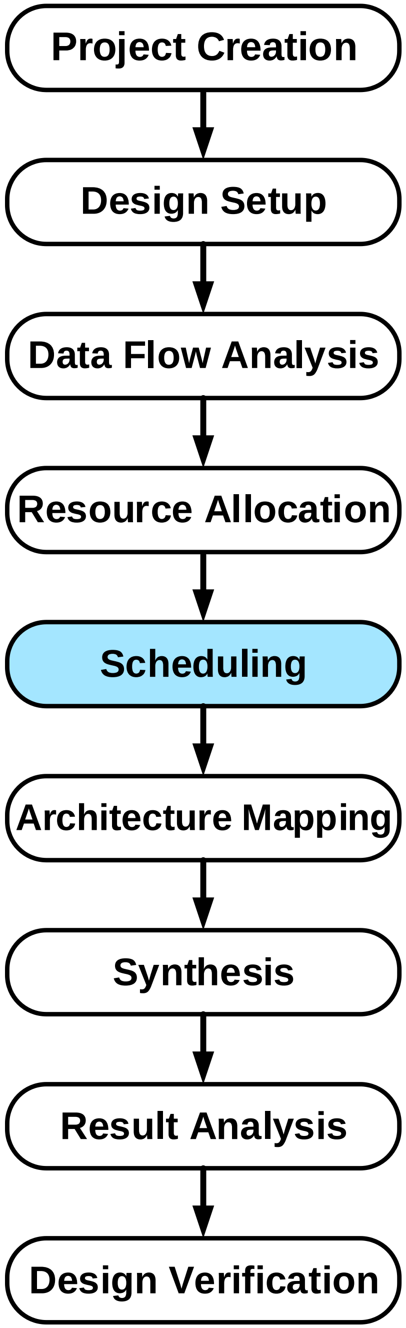
- Some operators may have multiple hardware resource implementations
  - Different area, delay, and latency specifications.

- Timing and area constraints will be applied to this implementation.

Resource Allocation

- All resources are selected from a technology specific pre-characterized library
    - Library is already determined in “Design Setup” step




Project Creation
Design Setup
Data Flow Analysis
Resource Allocation
Scheduling
Architecture Mapping
Synthesis
Result Analysis
Design Verification

Scheduling

- HLS adds "time" to the design during the "scheduling“ step.

- Scheduling determines that each operation in the DFG should be performed in which clock cycle.
  - So, based on a target clock frequency, registers should be added between operations.

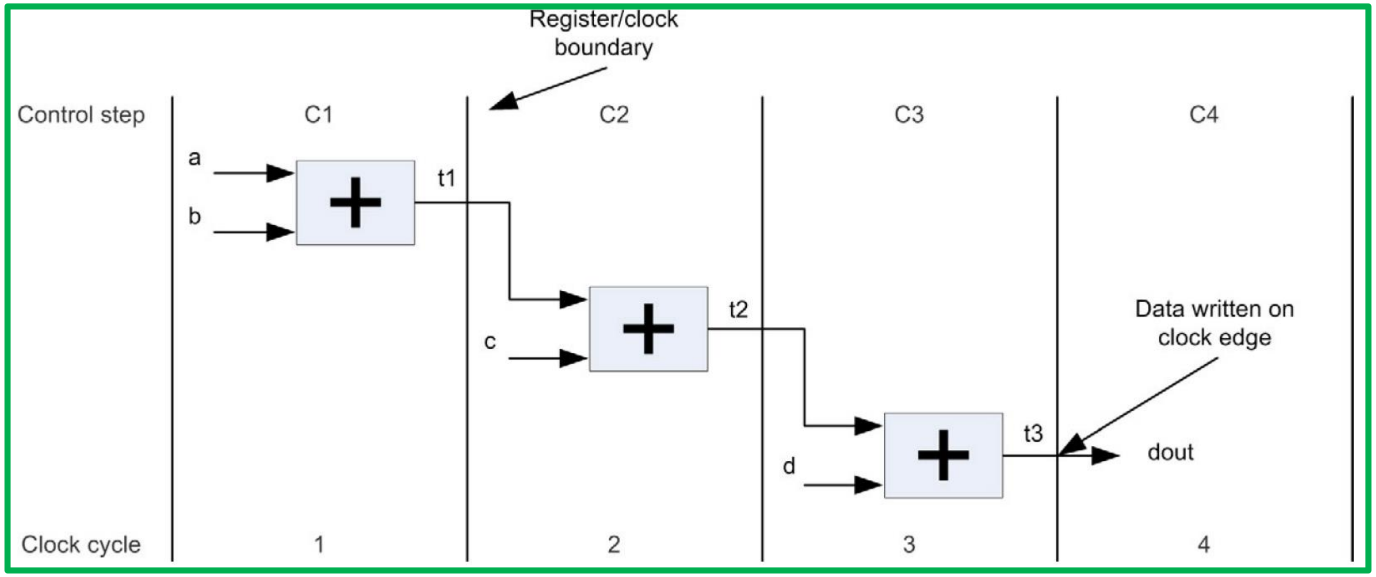
Scheduling

- This process is similar to technique what RTL designers would call **pipelining**
  - Inserting registers to reduce combinational delays

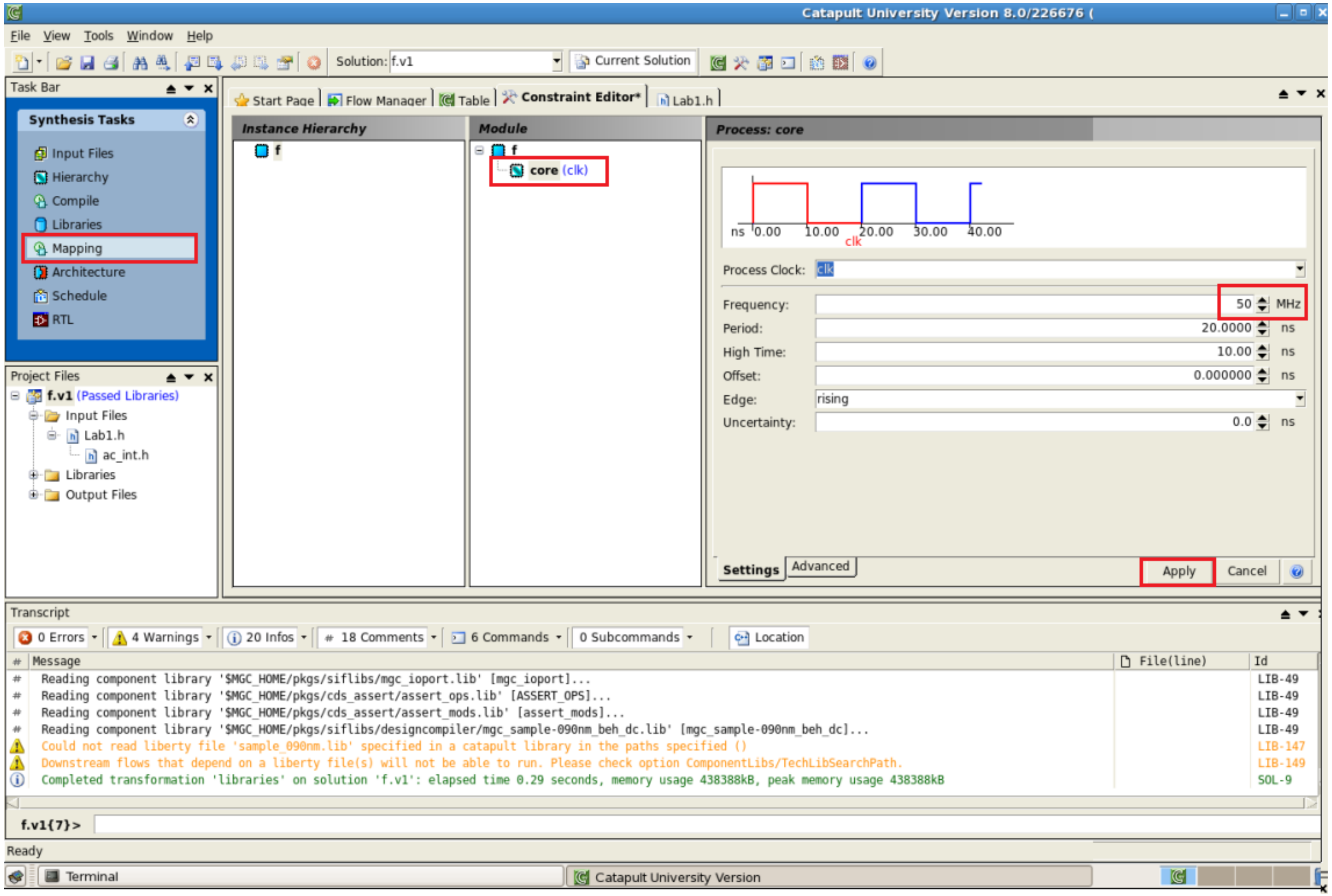
- Note that this is not the same as "**loop pipelining**" which is discussed later.

Scheduling

- Consider that in our design example:
  - "add" operation takes 3 ns
  - Clock period equals 5 ns
- Each add operation is scheduled in its own clock cycle C1, C2, and C3
  - Registers are inserted automatically between each adder.



Timing Constraints



f.v1 (Passed Libraries)

Process Clock: clk

Frequency: 50 MHz

Period: 20.0000 ns

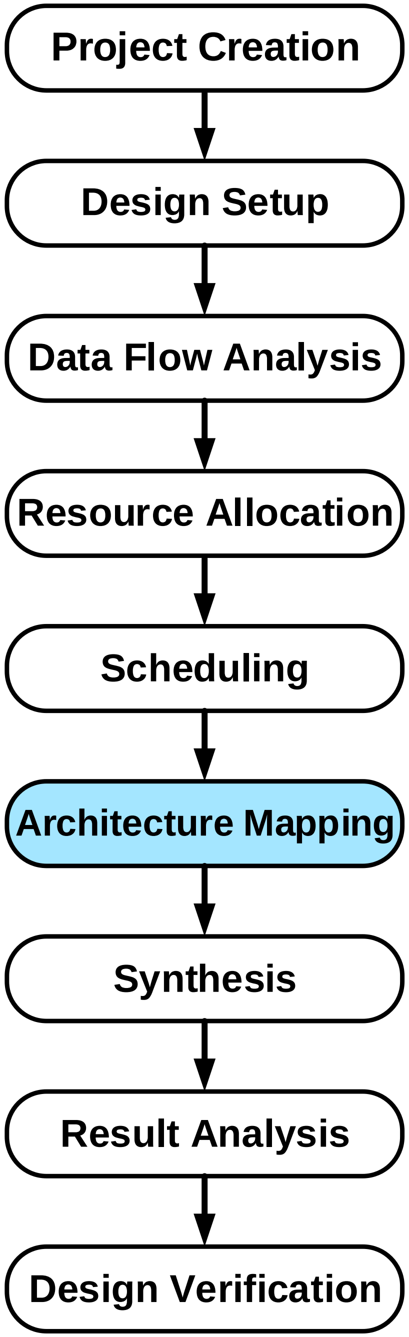
High Time: 10.00 ns

Offset: 0.000000 ns

Edge: rising

Uncertainty: 0.0 ns

```
#  Message                                                                                                File(line)  Id
#  Reading component library '$MGC_HOME/pkgs/siflibs/mgc_ioport.lib' [mgc_ioport]...                                 LIB-49
#  Reading component library '$MGC_HOME/pkgs/cds_assert/assert_ops.lib' [ASSERT_OPS]...                              LIB-49
#  Reading component library '$MGC_HOME/pkgs/cds_assert/assert_mods.lib' [assert_mods]...                             LIB-49
#  Reading component library '$MGC_HOME/pkgs/siflibs/designcompiler/mgc_sample-090nm_beh_dc.lib' [mgc_sample-090nm_beh_dc]...  LIB-49
⚠  Could not read liberty file 'sample_090nm.lib' specified in a catapult library in the paths specified ()      LIB-147
⚠  Downstream flows that depend on a liberty file(s) will not be able to run. Please check option ComponentLibs/TechLibSearchPath.  LIB-149
ⓘ  Completed transformation 'libraries' on solution 'f.v1': elapsed time 0.29 seconds, memory usage 438388kB, peak memory usage 438388kB  SOL-9

f.v1{7}>
Ready
```


Project Creation
Design Setup
Data Flow Analysis
Resource Allocation
Scheduling
Architecture Mapping
Synthesis
Result Analysis
Design Verification

Hardware Implementation

- The corresponding hardware that is generated from the schedule varies depending on how the design is constrained in terms of:
  - Resource allocation
  - The amount of loop pipelining

Loop Pipelining

- The top-level function call has an implied loop, also known as the main loop.
  - Each iteration of the implied loop corresponds to execution of the schedule

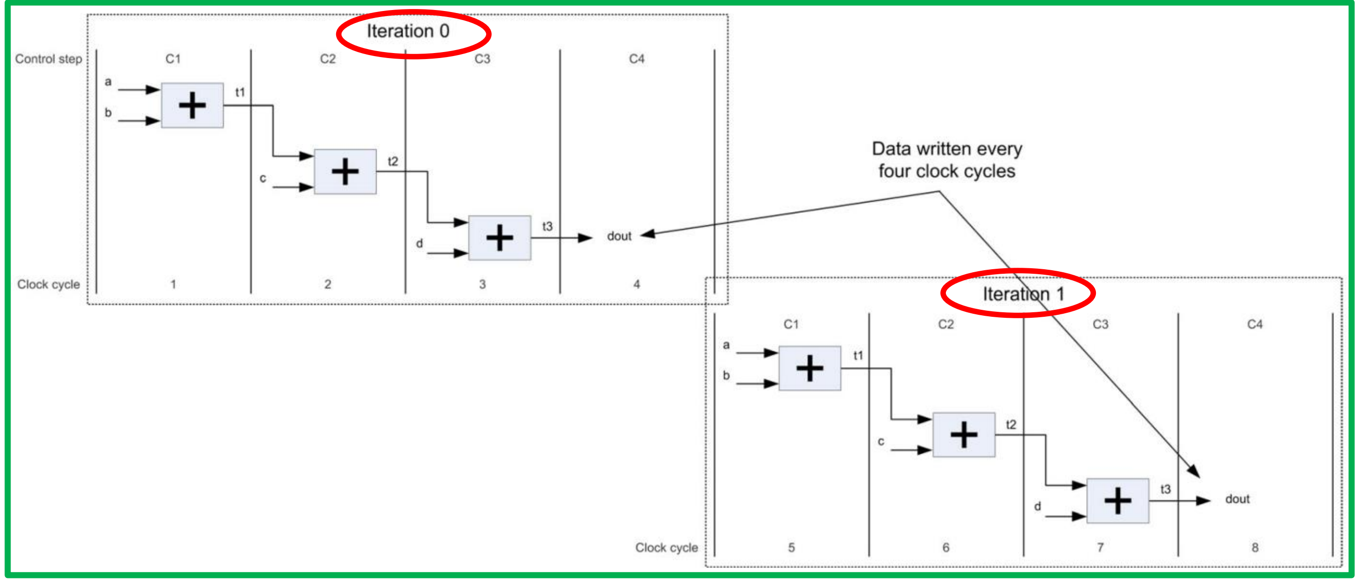
- "Loop Pipelining" allows a new iteration of a loop to be started before the current iteration has finished.
  - Execution of the loop iterations to be overlapped
  - Increasing the design performance by running loops in parallel

Initiation Interval (II)

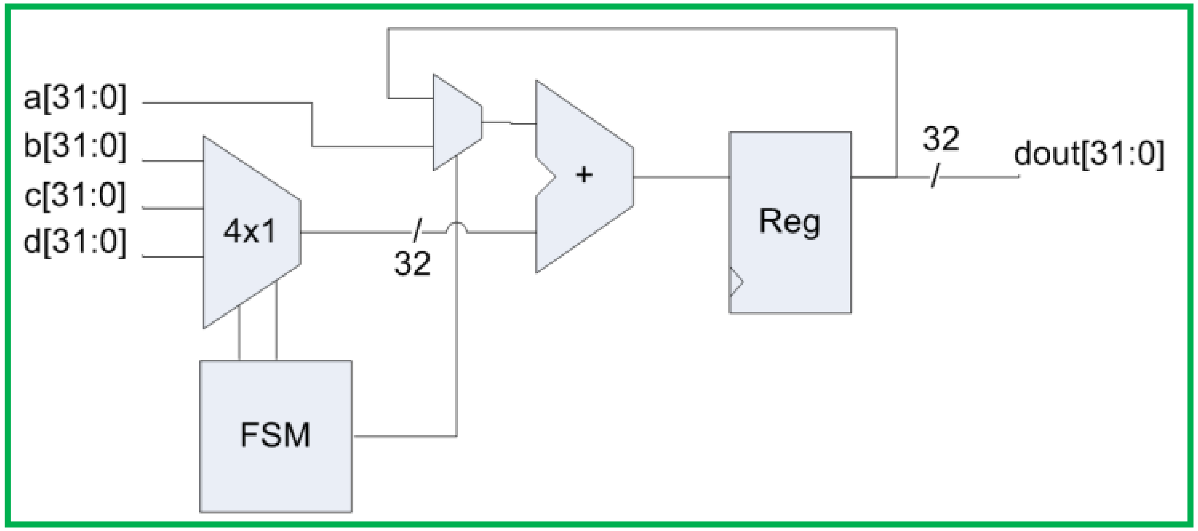
- The amount of overlap between the loops is determined by the "Initiation Interval (II)".
  - Specifies the number of pipeline stages

Case1: No Pipelining

- Design is left unconstrained
  - There is only a single pipeline stage
  - No overlap between execution of each iteration of main loop
- Data will be written every four clock cycles
  - Latency of three clock cycles
  - Throughput of four clock cycles



Case1: No Pipelining



- There is no overlap between any operation.
- Resulting hardware uses a single adder to accumulate a, b, c, and d.
- Reduction in the overall area.

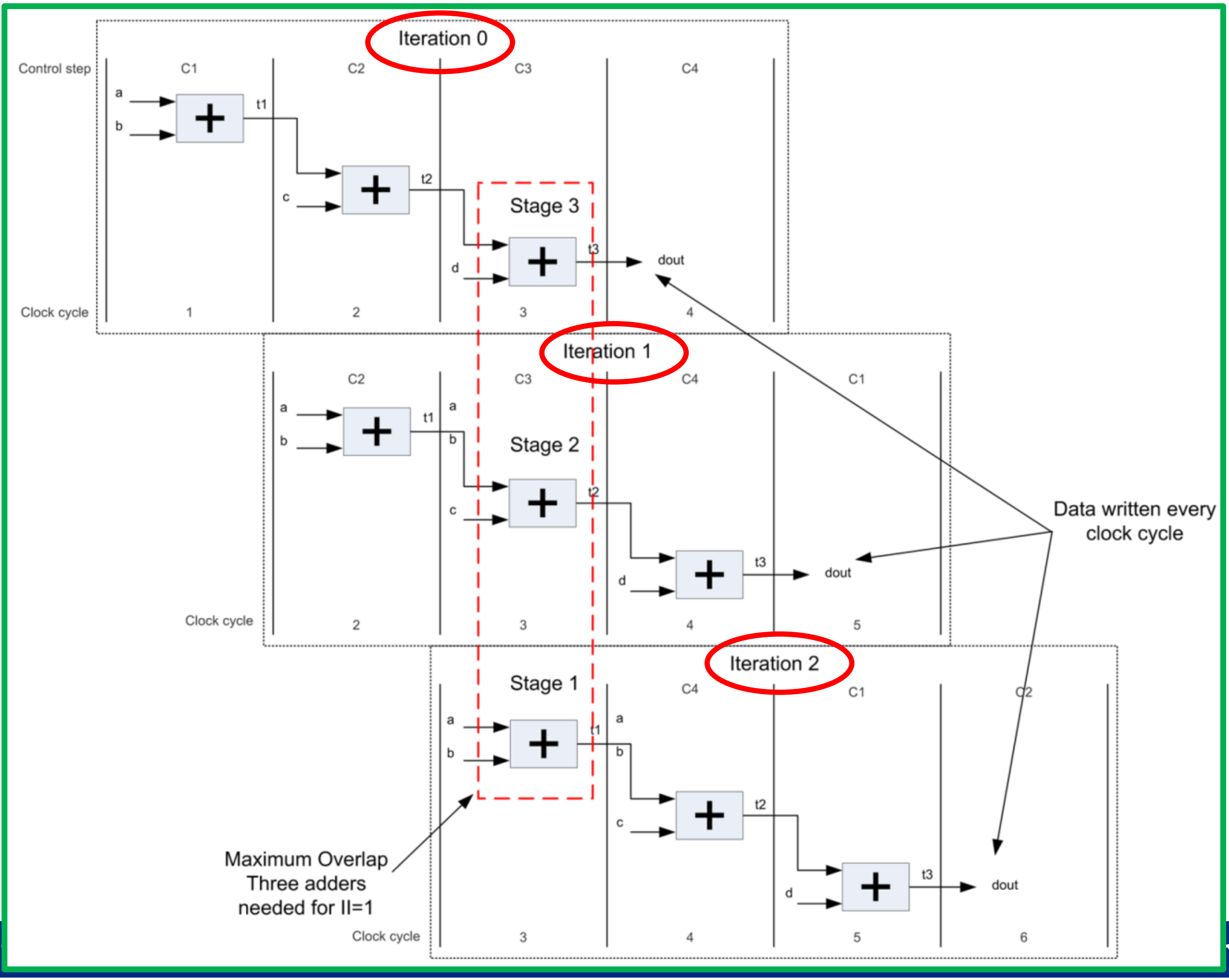
Case 2: Pipelining with II = 1

- ❑ Pipelining with an II=1 results in a new iteration started every clock cycle.
  - Latency of 3 clock cycles
  - Throughput of 1

- ❑Three adders are required in hardware since all three pipeline stages can be active at the same time.
  - Larger area

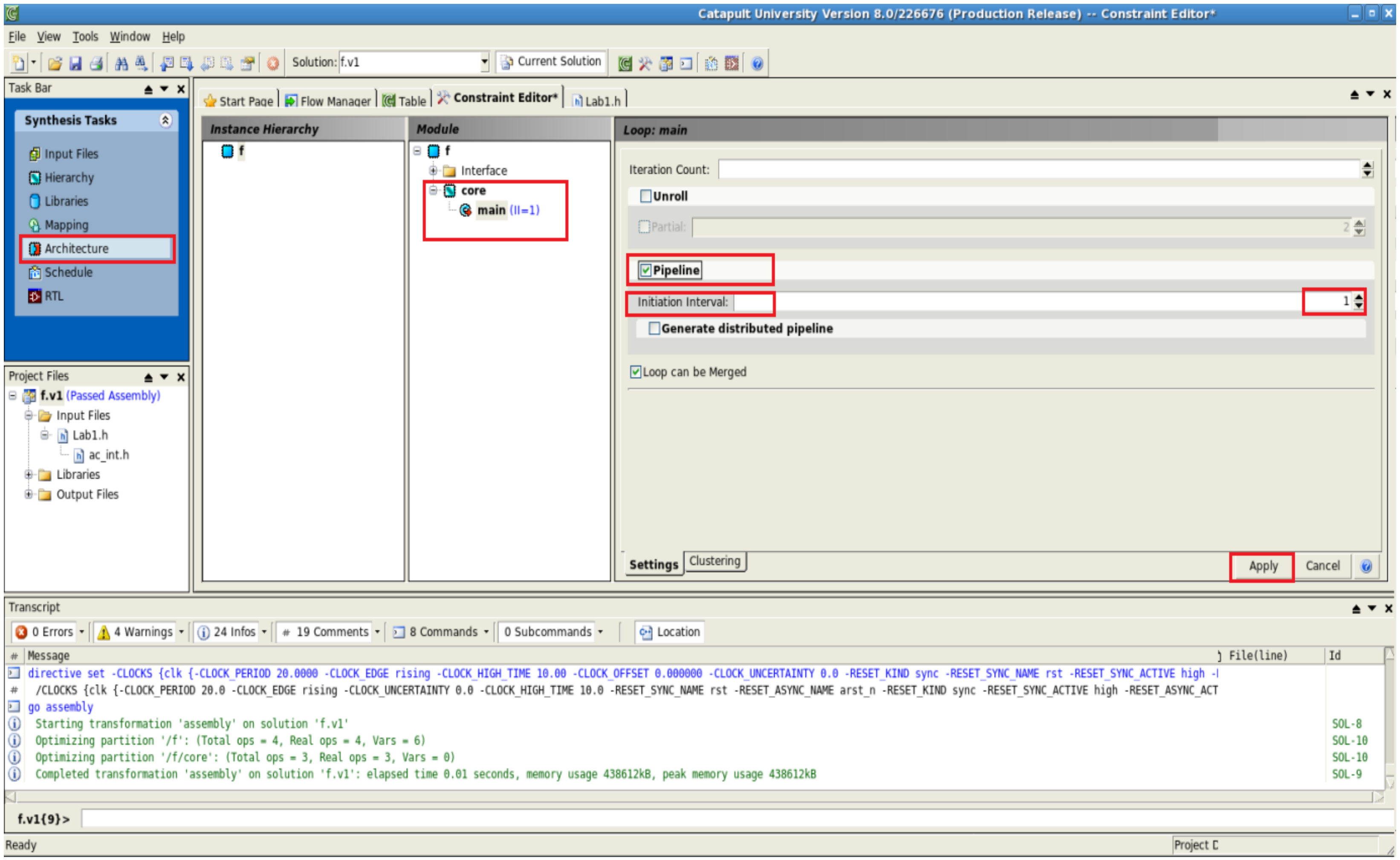
Case 2: Pipelining with II = 1

- Iteration one is started in C2 and iteration 2 is started in C3.



LUND UNIVERSITY

Architecture Specification



Iteration Count:

Unroll [ ]

Partial: 2

Pipeline [✓]

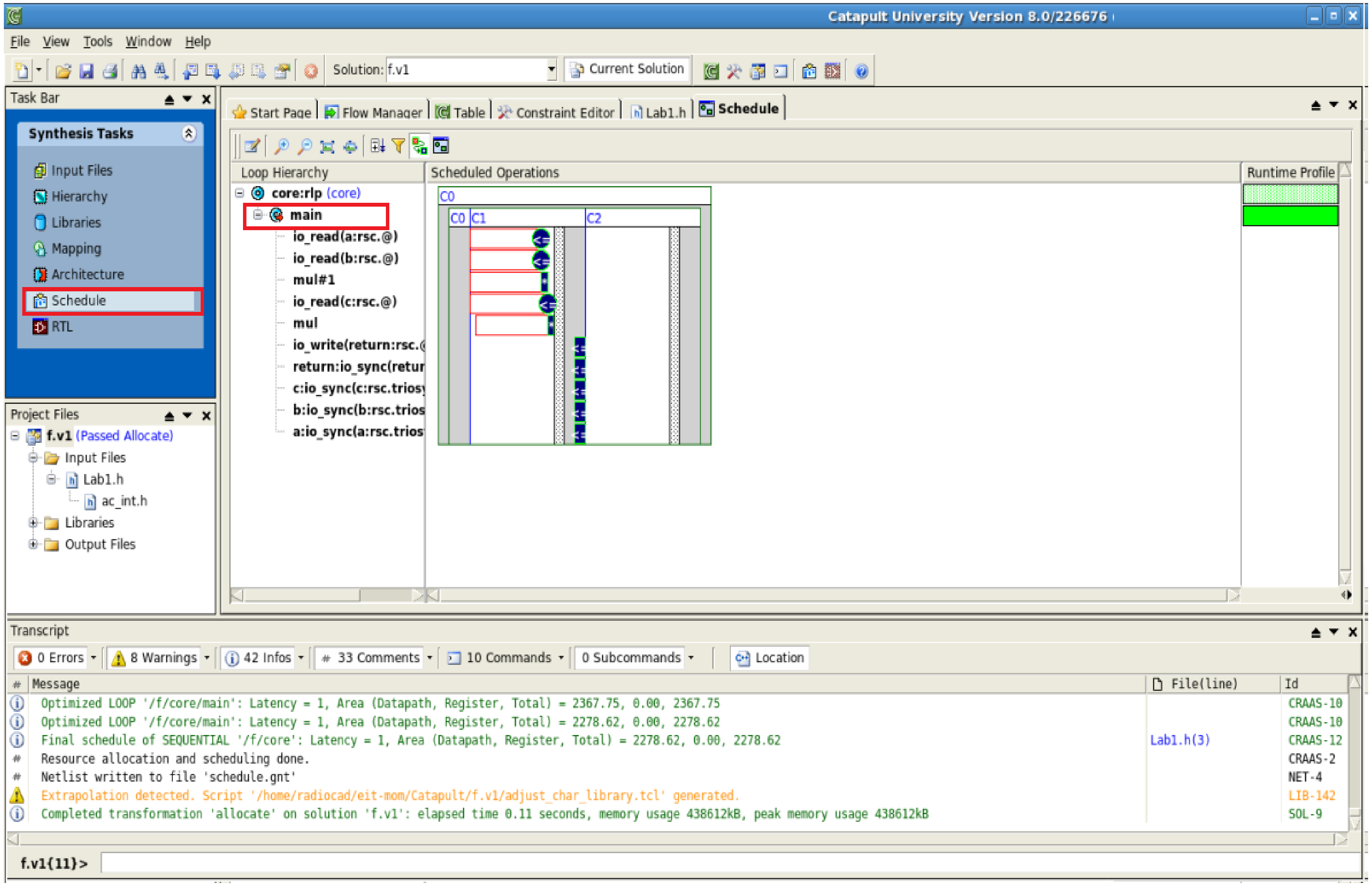
Initiation Interval: 1

Generate distributed pipeline [ ]

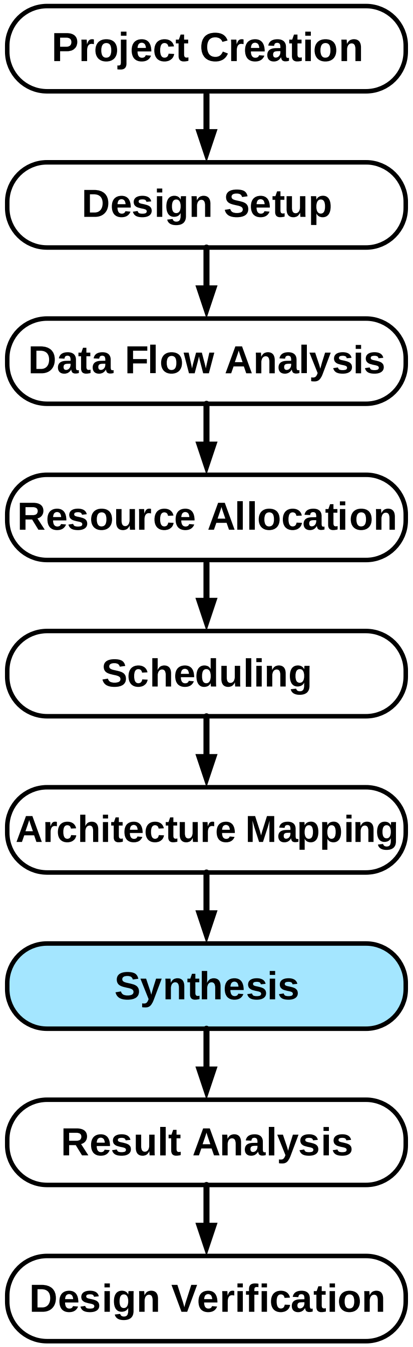
Loop can be Merged [✓]

```
#  Message                                                                                                                                 File(line)   Id
directive set -CLOCKS {clk {-CLOCK_PERIOD 20.0000 -CLOCK_EDGE rising -CLOCK_HIGH_TIME 10.00 -CLOCK_OFFSET 0.000000 -CLOCK_UNCERTAINTY 0.0 -RESET_KIND sync -RESET_SYNC_NAME rst -RESET_SYNC_ACTIVE high -I
#  /CLOCKS {clk {-CLOCK_PERIOD 20.0 -CLOCK_EDGE rising -CLOCK_UNCERTAINTY 0.0 -CLOCK_HIGH_TIME 10.0 -RESET_SYNC_NAME rst -RESET_ASYNC_NAME arst_n -RESET_KIND sync -RESET_SYNC_ACTIVE high -RESET_ASYNC_ACT
go assembly
   Starting transformation 'assembly' on solution 'f.v1'                                                                                             SOL-8
   Optimizing partition '/f': (Total ops = 4, Real ops = 4, Vars = 6)                                                                                SOL-10
   Optimizing partition '/f/core': (Total ops = 3, Real ops = 3, Vars = 0)                                                                           SOL-10
   Completed transformation 'assembly' on solution 'f.v1': elapsed time 0.01 seconds, memory usage 438612kB, peak memory usage 438612kB              SOL-9

f.v1{9}>
```

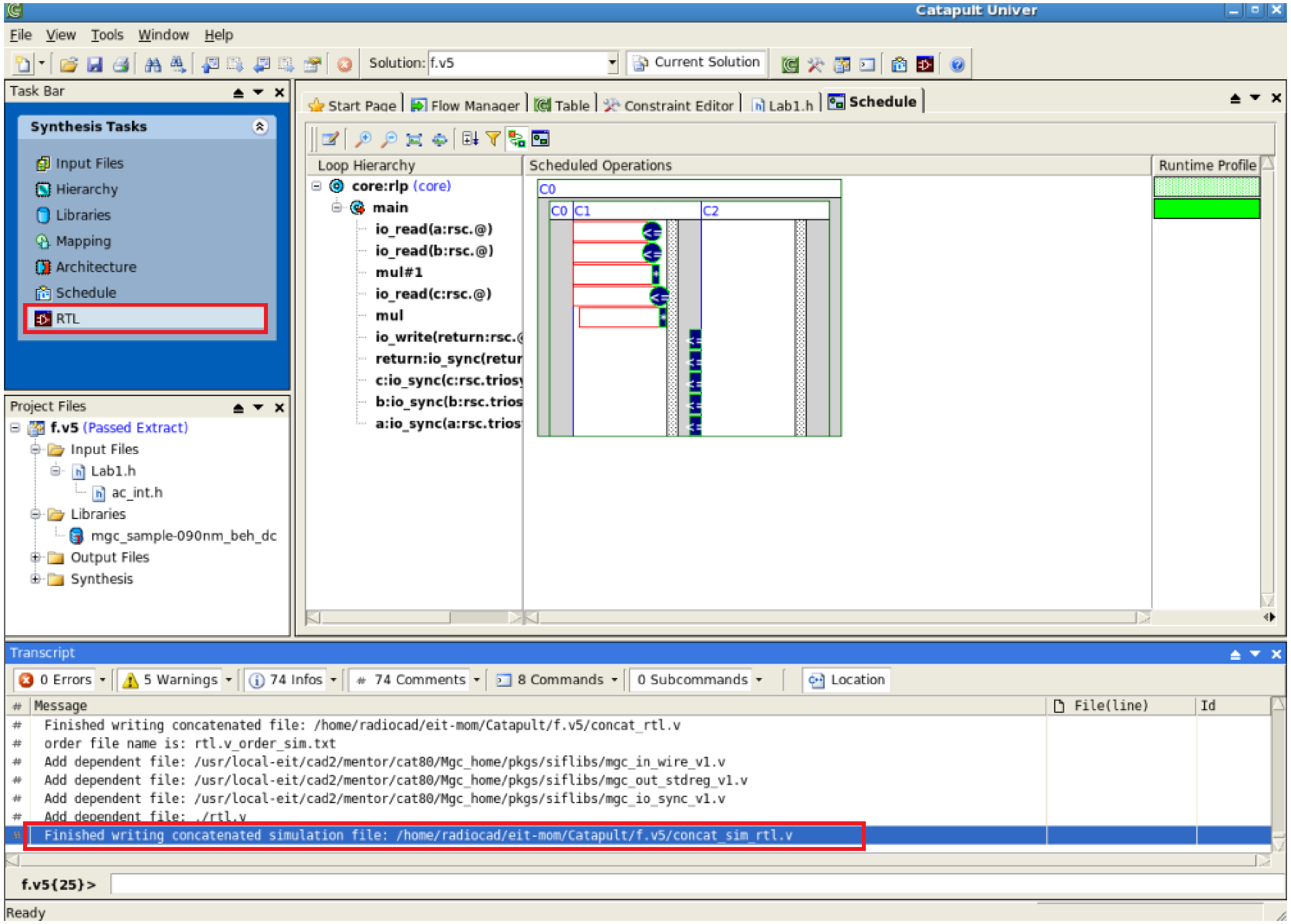
Design Schedule



```
#  Message                                                                                                                         File(line)   Id
ⓘ  Optimized LOOP '/f/core/main': Latency = 1, Area (Datapath, Register, Total) = 2367.75, 0.00, 2367.75                                          CRAAS-10
ⓘ  Optimized LOOP '/f/core/main': Latency = 1, Area (Datapath, Register, Total) = 2278.62, 0.00, 2278.62                                          CRAAS-10
ⓘ  Final schedule of SEQUENTIAL '/f/core': Latency = 1, Area (Datapath, Register, Total) = 2278.62, 0.00, 2278.62                     Lab1.h(3)    CRAAS-12
#  Resource allocation and scheduling done.                                                                                                       CRAAS-2
#  Netlist written to file 'schedule.gnt'                                                                                                         NET-4
⚠  Extrapolation detected. Script '/home/radiocad/eit-mom/Catapult/f.v1/adjust_char_library.tcl' generated.                                       LIB-142
ⓘ  Completed transformation 'allocate' on solution 'f.v1': elapsed time 0.11 seconds, memory usage 438612kB, peak memory usage 438612kB          SOL-9
```

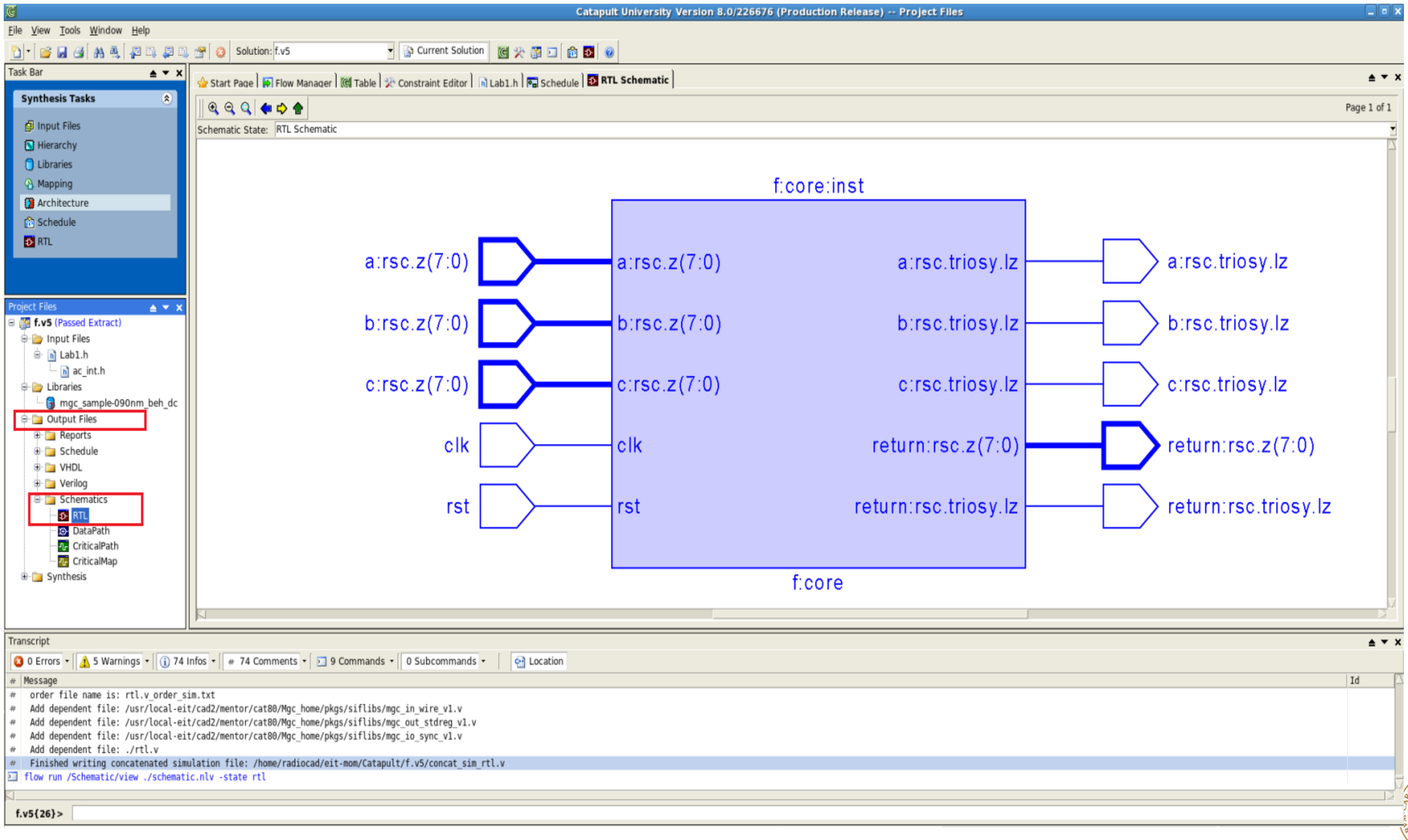

Project Creation
Design Setup
Data Flow Analysis
Resource Allocation
Scheduling
Architecture Mapping
Synthesis
Result Analysis
Design Verification

Design Synthesis



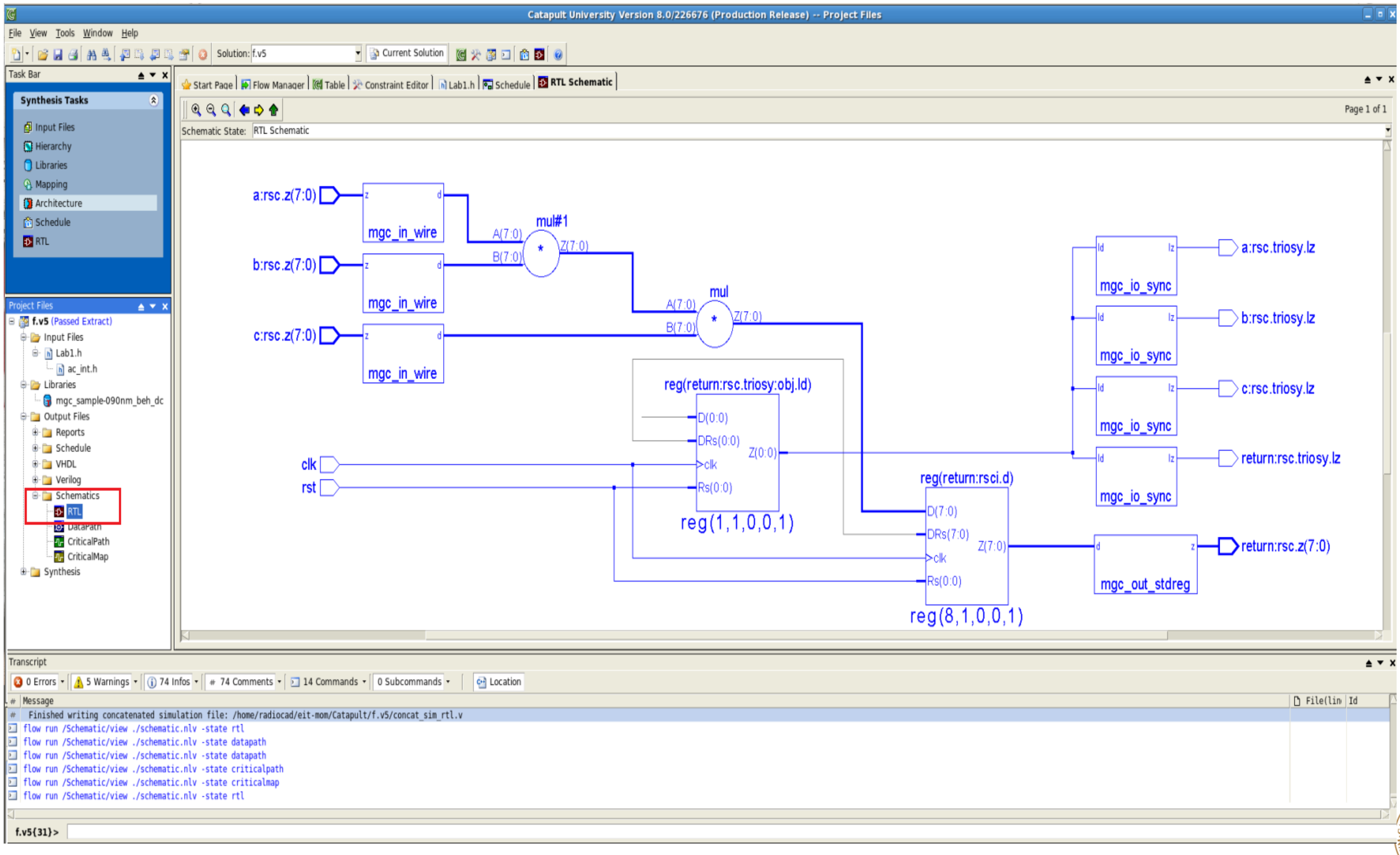
```
#  Finished writing concatenated file: /home/radiocad/eit-mom/Catapult/f.v5/concat_rtl.v
#  order file name is: rtl.v_order_sim.txt
#  Add dependent file: /usr/local-eit/cad2/mentor/cat80/Mgc_home/pkgs/siflibs/mgc_in_wire_v1.v
#  Add dependent file: /usr/local-eit/cad2/mentor/cat80/Mgc_home/pkgs/siflibs/mgc_out_stdreg_v1.v
#  Add dependent file: /usr/local-eit/cad2/mentor/cat80/Mgc_home/pkgs/siflibs/mgc_io_sync_v1.v
#  Add dependent file: ./rtl.v
#  Finished writing concatenated simulation file: /home/radiocad/eit-mom/Catapult/f.v5/concat_sim_rtl.v

f.v5{25}>
Ready
```

Design Schematic

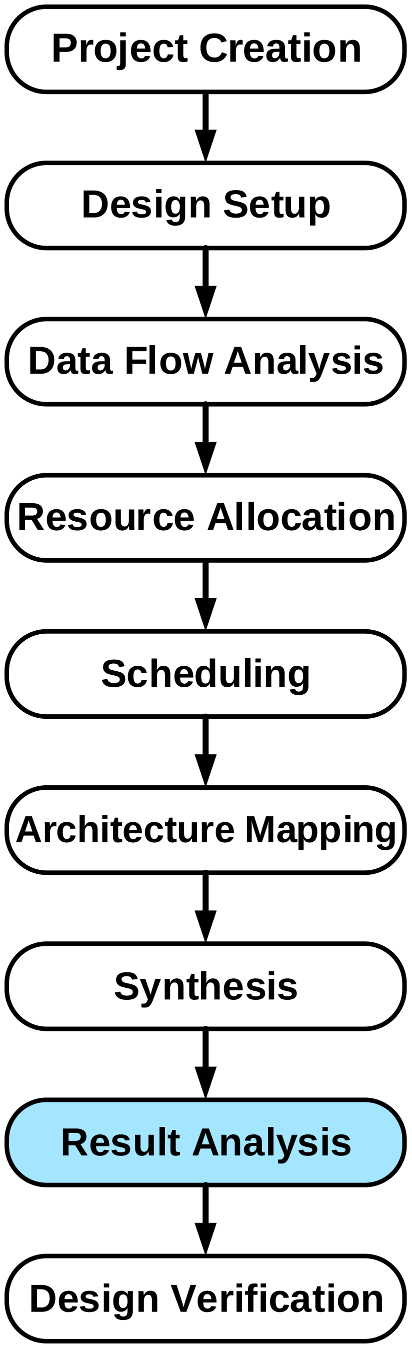


```
#  order file name is: rtl.v_order_sim.txt
#  Add dependent file: /usr/local-eit/cad2/mentor/cat80/Mgc_home/pkgs/siflibs/mgc_in_wire_v1.v
#  Add dependent file: /usr/local-eit/cad2/mentor/cat80/Mgc_home/pkgs/siflibs/mgc_out_stdreg_v1.v
#  Add dependent file: /usr/local-eit/cad2/mentor/cat80/Mgc_home/pkgs/siflibs/mgc_io_sync_v1.v
#  Add dependent file: ./rtl.v
#  Finished writing concatenated simulation file: /home/radiocad/eit-mom/Catapult/f.v5/concat_sim_rtl.v
   flow run /Schematic/view ./schematic.nlv -state rtl

f.v5{26}>
```

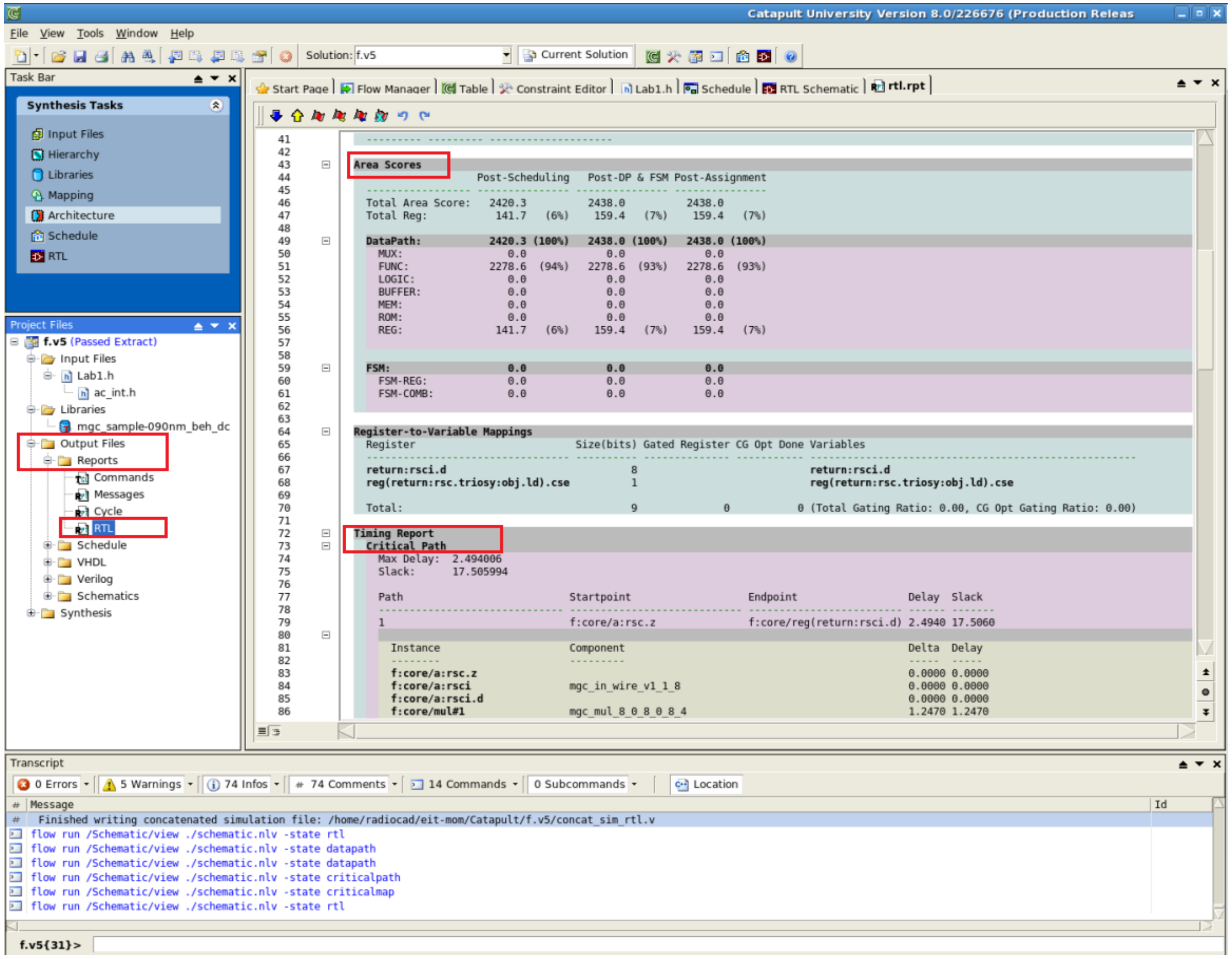
Design Schematic



```
# Finished writing concatenated simulation file: /home/radiocad/eit-mom/Catapult/f.v5/concat_sim_rtl.v
flow run /Schematic/view ./schematic.nlv -state rtl
flow run /Schematic/view ./schematic.nlv -state datapath
flow run /Schematic/view ./schematic.nlv -state datapath
flow run /Schematic/view ./schematic.nlv -state criticalpath
flow run /Schematic/view ./schematic.nlv -state criticalmap
flow run /Schematic/view ./schematic.nlv -state rtl
```


Project Creation
Design Setup
Data Flow Analysis
Resource Allocation
Scheduling
Architecture Mapping
Synthesis
Result Analysis
Design Verification

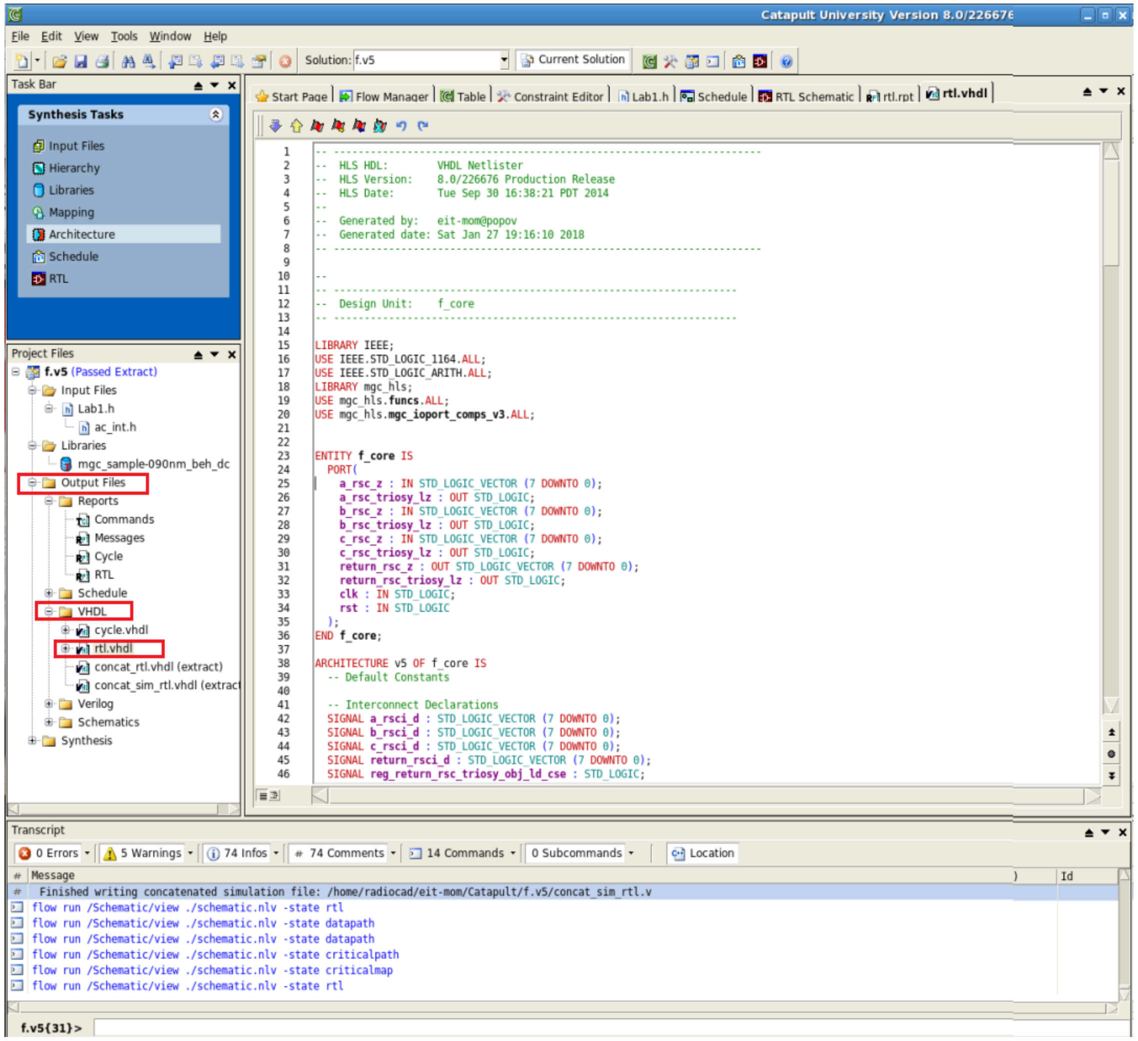
Result Analysis



```
  --------- --------- --------------------

Area Scores
                        Post-Scheduling   Post-DP & FSM Post-Assignment
  ----------------- ---------------- --------------- ---------------
  Total Area Score:   2420.3          2438.0          2438.0
  Total Reg:           141.7    (6%)   159.4    (7%)   159.4    (7%)

  DataPath:           2420.3 (100%)   2438.0 (100%)   2438.0 (100%)
    MUX:                 0.0             0.0             0.0
    FUNC:             2278.6   (94%)  2278.6   (93%)  2278.6   (93%)
    LOGIC:               0.0             0.0             0.0
    BUFFER:              0.0             0.0             0.0
    MEM:                 0.0             0.0             0.0
    ROM:                 0.0             0.0             0.0
    REG:               141.7    (6%)   159.4    (7%)   159.4    (7%)


  FSM:                   0.0             0.0             0.0
    FSM-REG:             0.0             0.0             0.0
    FSM-COMB:            0.0             0.0             0.0


Register-to-Variable Mappings
  Register                          Size(bits) Gated Register CG Opt Done Variables
  --------------------------------- ---------- -------------- ----------- -----------------------------------------------------
  return:rsci.d                              8                            return:rsci.d
  reg(return:rsc.triosy:obj.ld).cse          1                            reg(return:rsc.triosy:obj.ld).cse

  Total:                                     9              0           0 (Total Gating Ratio: 0.00, CG Opt Gating Ratio: 0.00)

Timing Report
  Critical Path
    Max Delay:  2.494006
    Slack:      17.505994

    Path                           Startpoint                  Endpoint                   Delay  Slack
    ------------------------------ --------------------------- -------------------------- ------ -------
    1                              f:core/a:rsc.z              f:core/reg(return:rsci.d)  2.4940 17.5060

      Instance                     Component                                              Delta  Delay
      --------                     ---------                                              -----  -----
      f:core/a:rsc.z                                                                      0.0000 0.0000
      f:core/a:rsci                mgc_in_wire_v1_1_8                                     0.0000 0.0000
      f:core/a:rsci.d                                                                     0.0000 0.0000
      f:core/mul#1                 mgc_mul_8_0_8_0_8_4                                    1.2470 1.2470
```

```
# Message                                                                                    Id
#  Finished writing concatenated simulation file: /home/radiocad/eit-mom/Catapult/f.v5/concat_sim_rtl.v
   flow run /Schematic/view ./schematic.nlv -state rtl
   flow run /Schematic/view ./schematic.nlv -state datapath
   flow run /Schematic/view ./schematic.nlv -state datapath
   flow run /Schematic/view ./schematic.nlv -state criticalpath
   flow run /Schematic/view ./schematic.nlv -state criticalmap
   flow run /Schematic/view ./schematic.nlv -state rtl

f.v5{31}>
```

HDL Generation

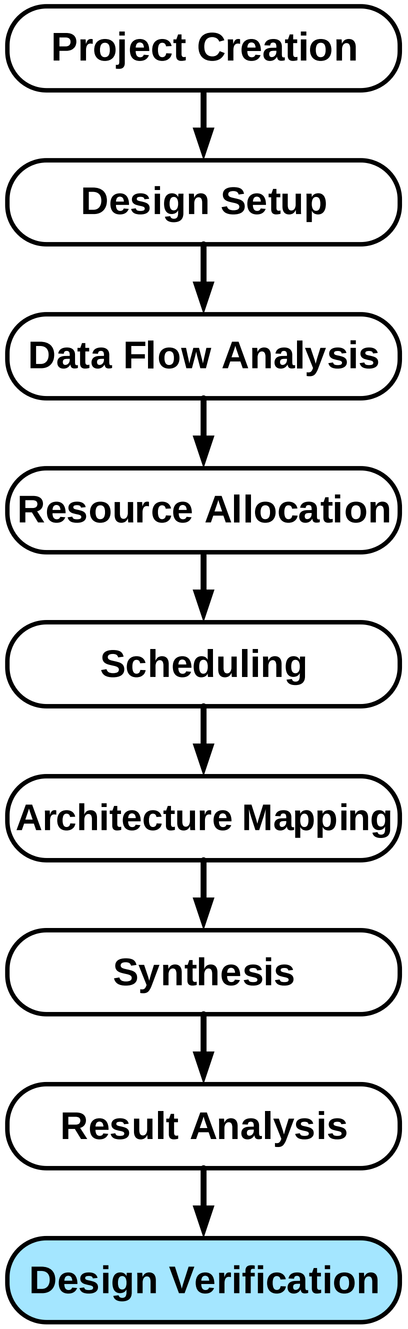


Catapult University Version 8.0/226676

```
-- ---------------------------------------------------------------------
--  HLS HDL:        VHDL Netlister
--  HLS Version:    8.0/226676 Production Release
--  HLS Date:       Tue Sep 30 16:38:21 PDT 2014
--
--  Generated by:   eit-mom@popov
--  Generated date: Sat Jan 27 19:16:10 2018
-- ---------------------------------------------------------------------


--
-- ------------------------------------------------------------------
--  Design Unit:    f_core
-- ------------------------------------------------------------------

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
LIBRARY mgc_hls;
USE mgc_hls.funcs.ALL;
USE mgc_hls.mgc_ioport_comps_v3.ALL;


ENTITY f_core IS
  PORT(
    a_rsc_z : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
    a_rsc_triosy_lz : OUT STD_LOGIC;
    b_rsc_z : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
    b_rsc_triosy_lz : OUT STD_LOGIC;
    c_rsc_z : IN STD_LOGIC_VECTOR (7 DOWNTO 0);
    c_rsc_triosy_lz : OUT STD_LOGIC;
    return_rsc_z : OUT STD_LOGIC_VECTOR (7 DOWNTO 0);
    return_rsc_triosy_lz : OUT STD_LOGIC;
    clk : IN STD_LOGIC;
    rst : IN STD_LOGIC
  );
END f_core;

ARCHITECTURE v5 OF f_core IS
  -- Default Constants

  -- Interconnect Declarations
  SIGNAL a_rsci_d : STD_LOGIC_VECTOR (7 DOWNTO 0);
  SIGNAL b_rsci_d : STD_LOGIC_VECTOR (7 DOWNTO 0);
  SIGNAL c_rsci_d : STD_LOGIC_VECTOR (7 DOWNTO 0);
  SIGNAL return_rsci_d : STD_LOGIC_VECTOR (7 DOWNTO 0);
  SIGNAL reg_return_rsc_triosy_obj_ld_cse : STD_LOGIC;
```

Transcript

```
 Finished writing concatenated simulation file: /home/radiocad/eit-mom/Catapult/f.v5/concat_sim_rtl.v
flow run /Schematic/view ./schematic.nlv -state rtl
flow run /Schematic/view ./schematic.nlv -state datapath
flow run /Schematic/view ./schematic.nlv -state datapath
flow run /Schematic/view ./schematic.nlv -state criticalpath
flow run /Schematic/view ./schematic.nlv -state criticalmap
flow run /Schematic/view ./schematic.nlv -state rtl
f.v5{31}>
```


Project Creation
Design Setup
Data Flow Analysis
Resource Allocation
Scheduling
Architecture Mapping
Synthesis
Result Analysis
Design Verification

Test Bench

```
#include "ac_int.h";
#include <iostream>
#include <fstream>
// Include the C/C++ function header
#include "lab1.h"
// Include the SCVerify header
#include "mc_testbench.h"
int main(int argc, char *argv[]){
  ac_int<8> a = 1;
  ac_int<8> b = 2;
  ac_int<8> result = 0;
  // Test simuli. Five iterations.
  for (int s_idx = 0; s_idx < 5; s_idx++) {
    result = f(a,b,s_idx);
    // Generate some output
    std::cout << "a*b*c = result: " << a.to_int() << "*"
<< b.to_int() << "*"<< s_idx << " = " << result.to_int()
<< std::endl;
  }
return 0;
}
```

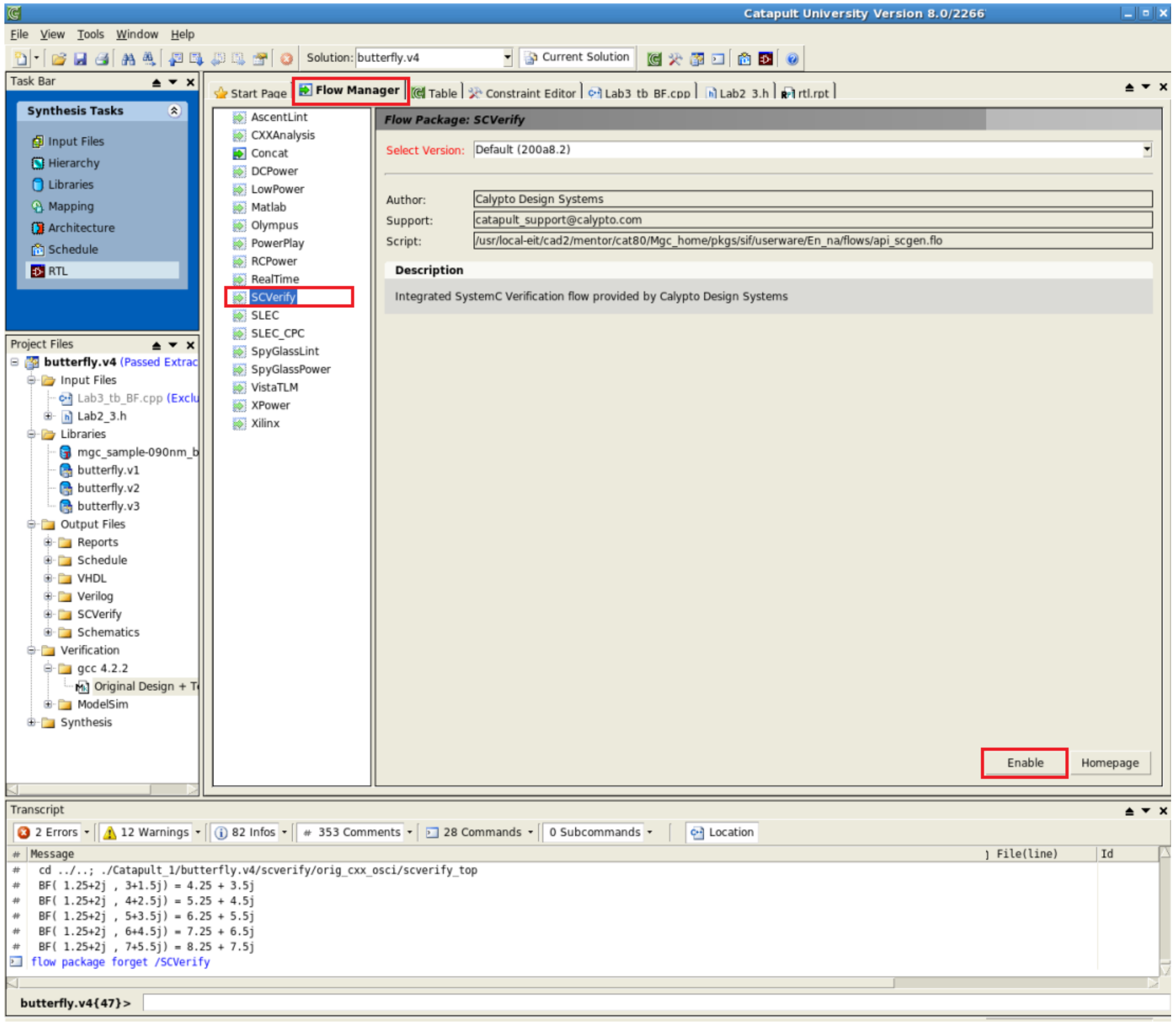
- Include your design source file
- Call your design
- Generate inputs to the design
- Write outputs in the console

Design Verification

- **SCVerify** flow in Catapult automatically generates the verification infrastructure.
    - The functionality of the generated RTL against the users original source code will be verified.

- SCVerify supports Mentor QuestaSim/ModelSim, Synopsys VCS and Cadence IUS/NCSim simulation environments.

Design Verification



Solution: butterfly.v4 | Current Solution

Flow Package: SCVerify

Select Version: Default (200a8.2)

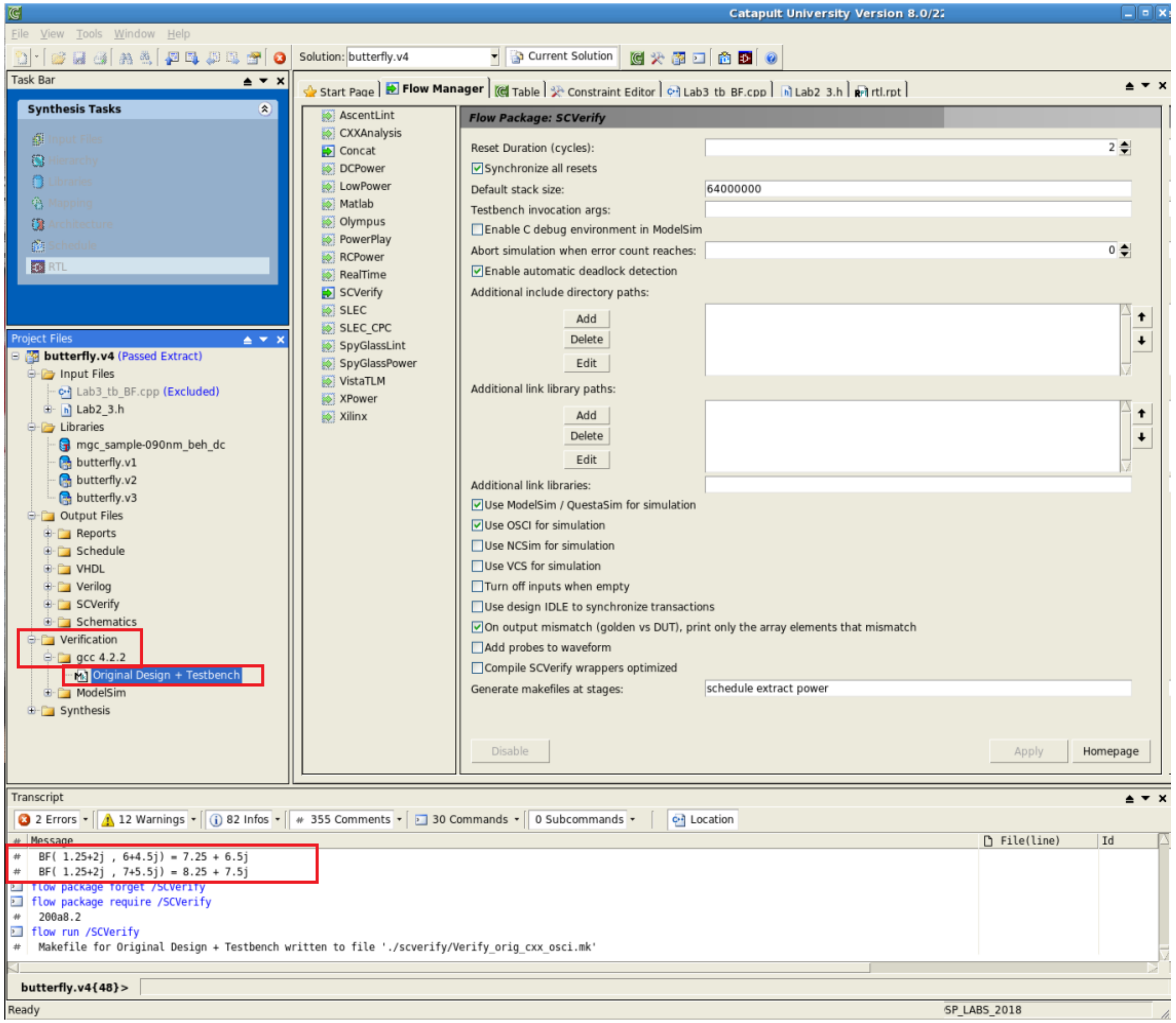
Author: Calypto Design Systems
Support: catapult_support@calypto.com
Script: /usr/local-eit/cad2/mentor/cat80/Mgc_home/pkgs/sif/userware/En_na/flows/api_scgen.flo

Description

Integrated SystemC Verification flow provided by Calypto Design Systems

```
#  Message                                                              File(line)   Id
#   cd ../..; ./Catapult_1/butterfly.v4/scverify/orig_cxx_osci/scverify_top
#   BF( 1.25+2j , 3+1.5j) = 4.25 + 3.5j
#   BF( 1.25+2j , 4+2.5j) = 5.25 + 4.5j
#   BF( 1.25+2j , 5+3.5j) = 6.25 + 5.5j
#   BF( 1.25+2j , 6+4.5j) = 7.25 + 6.5j
#   BF( 1.25+2j , 7+5.5j) = 8.25 + 7.5j
    flow package forget /SCVerify

butterfly.v4{47}>
```

Design Verification



Flow Package: SCVerify

- Reset Duration (cycles): 2
- [✓] Synchronize all resets
- Default stack size: 64000000
- Testbench invocation args:
- [ ] Enable C debug environment in ModelSim
- Abort simulation when error count reaches: 0
- [✓] Enable automatic deadlock detection
- Additional include directory paths:
- Additional link library paths:
- Additional link libraries:
- [✓] Use ModelSim / QuestaSim for simulation
- [✓] Use OSCI for simulation
- [ ] Use NCSim for simulation
- [ ] Use VCS for simulation
- [ ] Turn off inputs when empty
- [ ] Use design IDLE to synchronize transactions
- [✓] On output mismatch (golden vs DUT), print only the array elements that mismatch
- [ ] Add probes to waveform
- [ ] Compile SCVerify wrappers optimized
- Generate makefiles at stages: schedule extract power

```
#  BF( 1.25+2j , 6+4.5j) = 7.25 + 6.5j
#  BF( 1.25+2j , 7+5.5j) = 8.25 + 7.5j
   flow package forget /SCVerify
   flow package require /SCVerify
#  200a8.2
   flow run /SCVerify
#  Makefile for Original Design + Testbench written to file './scverify/Verify_orig_cxx_osci.mk'

butterfly.v4{48}>
```

References

❑High-Level Synthesis, Blue Book, Mentor Graphics Corporation

❑Algorthmic C™ Datatypes, Calypto Design Systems, Inc

LUND
UNIVERSITY