Traineeships in Advanced Computing for High Energy Physics (TAC-HEP)

GPU & FPGA module training: Part-2

Week-4: Vivado HLS: *Pragma's effect on performance*

Lecture-8: April 12th 2023

Varun Sharma

University of Wisconsin – Madison, USA

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

So Far…

- **FPGA and its architecture**
  - Registor/Flip-Flops, LUTs/Logic Cells, DSP, BRAMs
  - Clock Frequency, Latency
  - Extracting control logic & Implementing I/O ports
- **Parallelism in FPGA**
  - Scheduling, Pipelining, DataFlow
- **Vivado HLS**
  - Introduction, Setup, Hands-on for GUI/CLI, Introduction to Pragmas

Today:

- Continue with Pragmas and their effects on performance

TAC-HEP 2023

HLS Pragmas Effect [Ref]

Performance in term of Resource utilization and timing (latency)

Pragmas by type

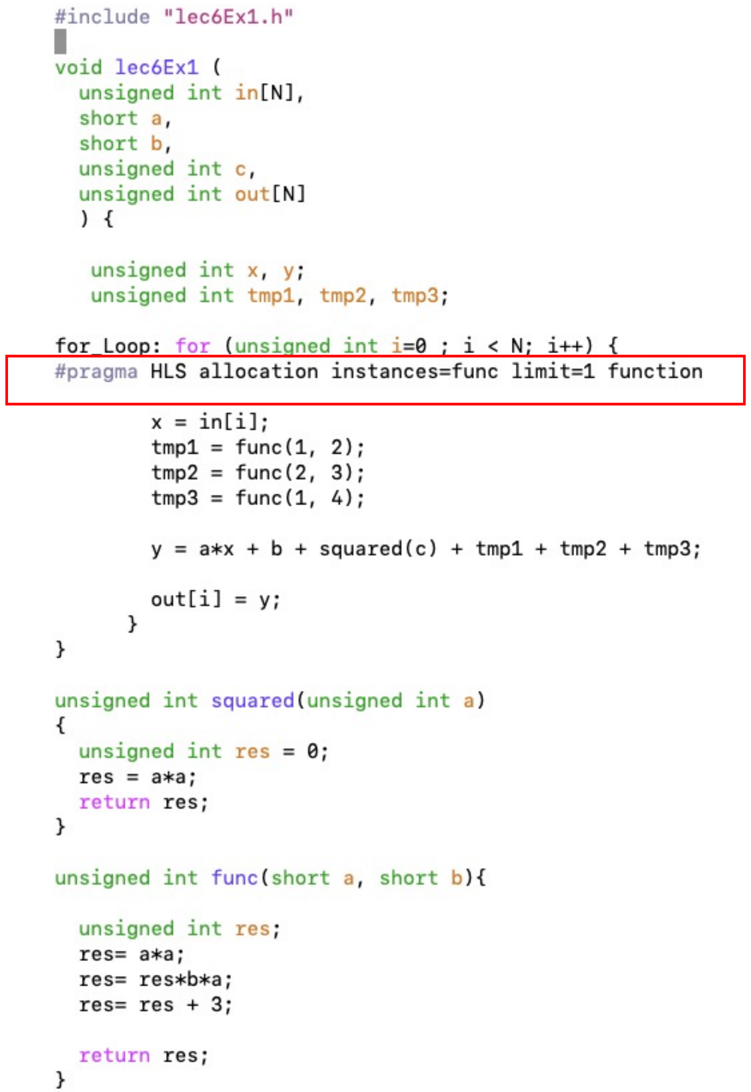
Type	Attributes	
Kernel Optimization	pragma HLS allocation pragma HLS expression_balance pragma HLS latency	pragma HLS reset pragma HLS resource pragma HLS stable
Function Inlining	pragma HLS inline pragma HLS function_instantiate	
Interface Synthesis	pragma HLS interface	
Task-level Pipeline	pragma HLS dataflow pragma HLS stream	
Pipeline	pragma HLS pipeline pragma HLS occurrence	
Loop Unrolling	pragma HLS unroll pragma HLS dependence	
Loop Optimization	pragma HLS loop_flatten pragma HLS loop_merge	pragma HLS loop_tripcount
Array Optimization	pragma HLS array_map pragma HLS array_partition	pragma HLS array_reshape
Structure Packing	pragma HLS data_pack	

Pragma HLS allocation

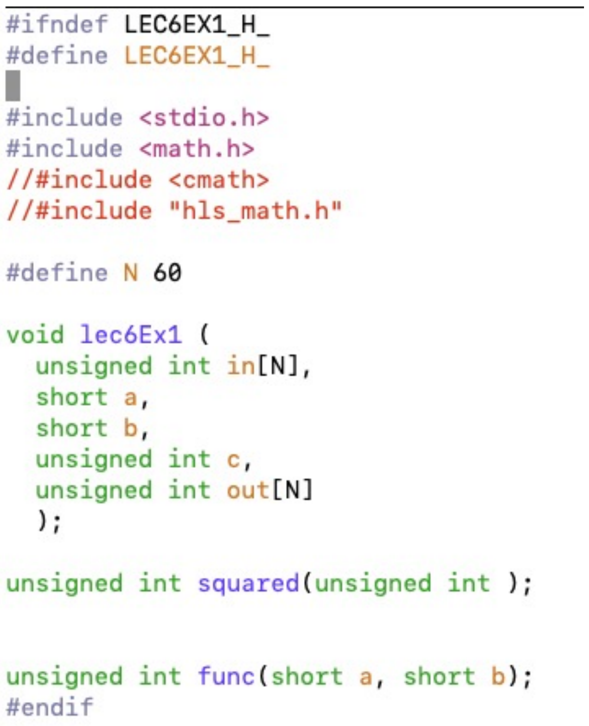
#pragma HLS allocation instances=<list> limit=<value> <type>

- **Instance<list>*:** Name of the function, operator, or cores
- **limit=<value>*:** Specifies the limit of instances to be used in kernel
- **<type>*:** Specifies the allocation applies to a function, an operator or a core (hardware component) used to create the design (such as adder, multiplier, BRAM)
  - *Function:* allocation applies to the functions listed in the instances=
  - *Operation:* applies to the operations listed in the instances=
  - *Core:* applies to the cores

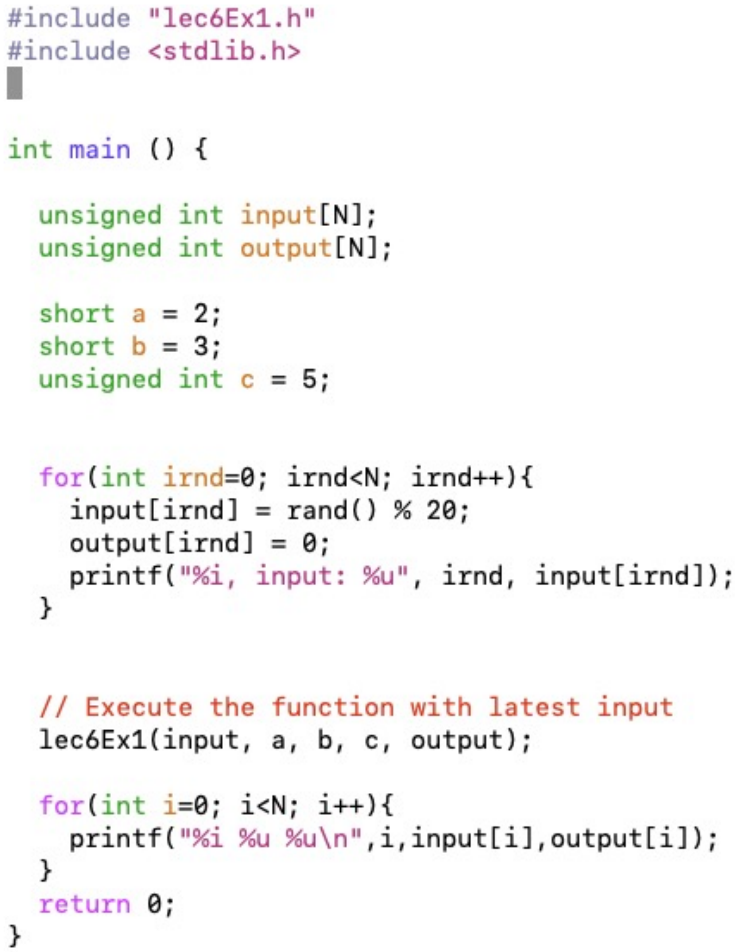
Example



```
#include "lec6Ex1.h"

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  ) {

   unsigned int x, y;
   unsigned int tmp1, tmp2, tmp3;

for_Loop: for (unsigned int i=0 ; i < N; i++) {
#pragma HLS allocation instances=func limit=1 function

        x = in[i];
        tmp1 = func(1, 2);
        tmp2 = func(2, 3);
        tmp3 = func(1, 4);

        y = a*x + b + squared(c) + tmp1 + tmp2 + tmp3;

        out[i] = y;
      }
}

unsigned int squared(unsigned int a)
{
  unsigned int res = 0;
  res = a*a;
  return res;
}

unsigned int func(short a, short b){

  unsigned int res;
  res= a*a;
  res= res*b*a;
  res= res + 3;

  return res;
}
```



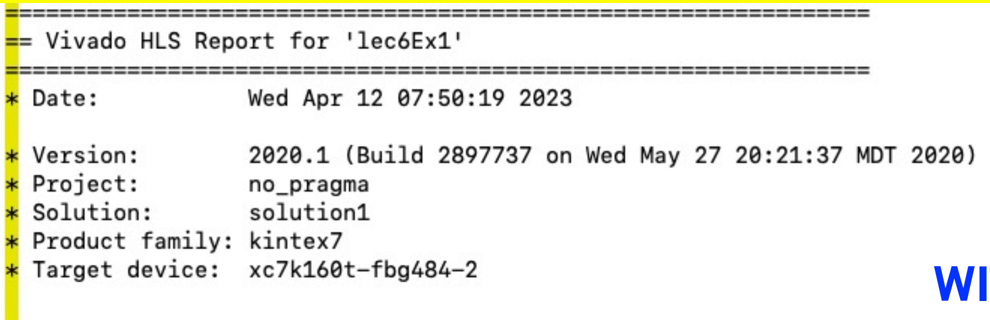
```
#ifndef LEC6EX1_H_
#define LEC6EX1_H_

#include <stdio.h>
#include <math.h>
//#include <cmath>
//#include "hls_math.h"

#define N 60

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  );

unsigned int squared(unsigned int );


unsigned int func(short a, short b);
#endif
```



```
#include "lec6Ex1.h"
#include <stdlib.h>

int main () {

  unsigned int input[N];
  unsigned int output[N];

  short a = 2;
  short b = 3;
  unsigned int c = 5;


  for(int irnd=0; irnd<N; irnd++){
    input[irnd] = rand() % 20;
    output[irnd] = 0;
    printf("%i, input: %u", irnd, input[irnd]);
  }


  // Execute the function with latest input
  lec6Ex1(input, a, b, c, output);

  for(int i=0; i<N; i++){
    printf("%i %u %u\n",i,input[i],output[i]);
  }
  return 0;
}
```

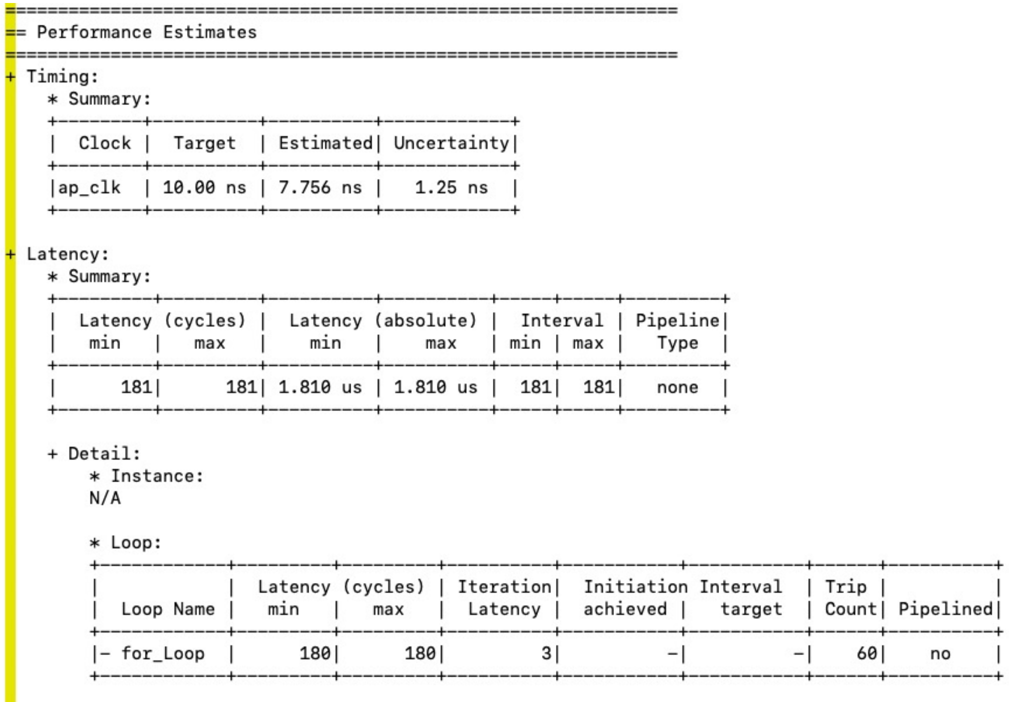
Pragma HLS allocation

#pragma HLS allocation instances=<list> limit=<value> <type>



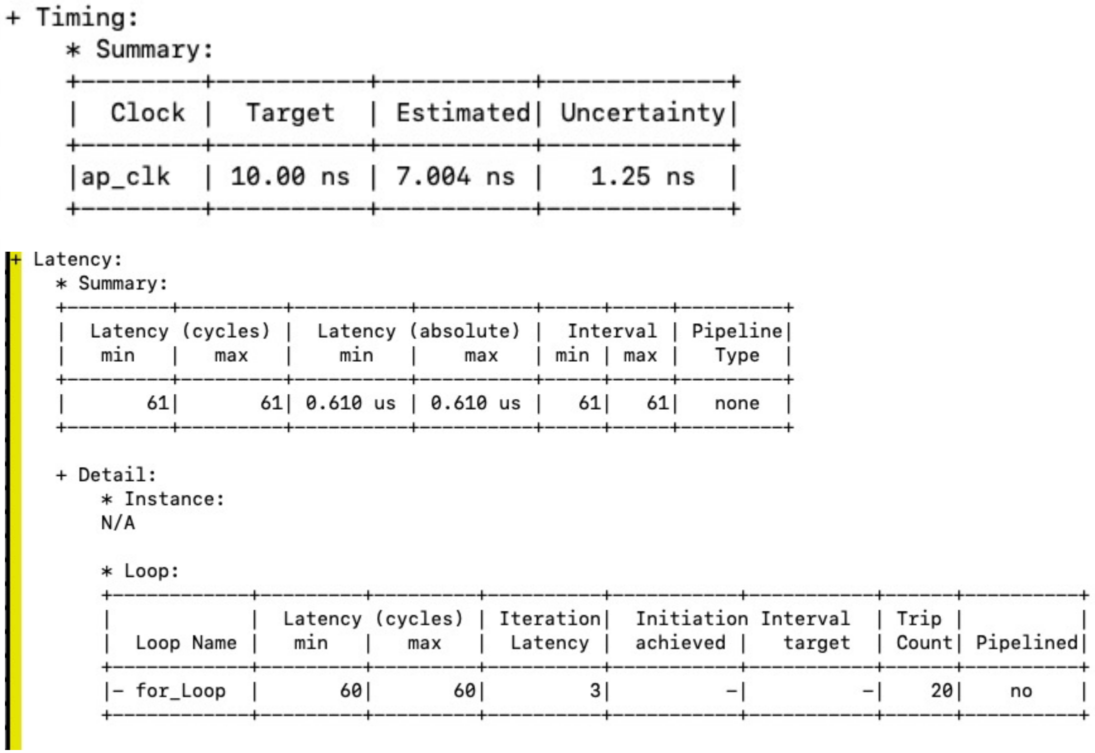
```
================================================================
== Vivado HLS Report for 'lec6Ex1'
================================================================
* Date:           Wed Apr 12 07:50:19 2023

* Version:        2020.1 (Build 2897737 on Wed May 27 20:21:37 MDT 2020)
* Project:        no_pragma
* Solution:       solution1
* Product family: kintex7
* Target device:  xc7k160t-fbg484-2
```

WITHOUT PRAGMA



```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+-----------+-----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute)   |  Interval | Pipeline|
    |   min   |   max   |    min    |    max    | min | max |   Type  |
    +---------+---------+-----------+-----------+-----+-----+---------+
    |      181|      181| 1.810 us  | 1.810 us  |  181|  181|   none  |
    +---------+---------+-----------+-----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |- for_Loop  |      180|      180|         3|          -|          -|    60|    no    |
        +------------+---------+---------+----------+-----------+-----------+------+----------+
```

WITH PRAGMA

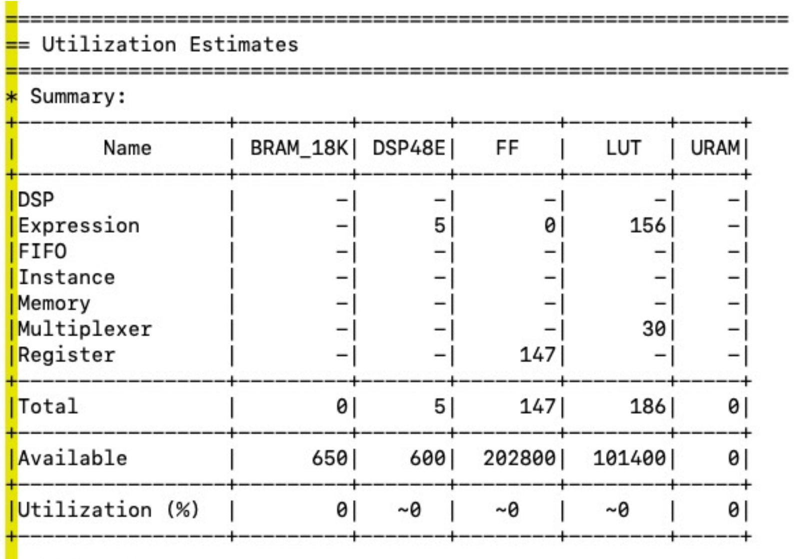


```
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 7.004 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+-----------+-----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute)   |  Interval | Pipeline|
    |   min   |   max   |    min    |    max    | min | max |   Type  |
    +---------+---------+-----------+-----------+-----+-----+---------+
    |       61|       61| 0.610 us  | 0.610 us  |   61|   61|   none  |
    +---------+---------+-----------+-----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |- for_Loop  |       60|       60|         3|          -|          -|    20|    no    |
        +------------+---------+---------+----------+-----------+-----------+------+----------+
```

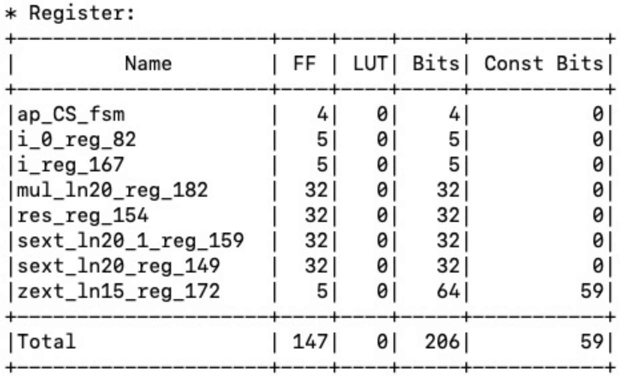
Pragma HLS allocation

#pragma HLS allocation instances=<list> limit=<value> <type>

WITHOUT PRAGMA

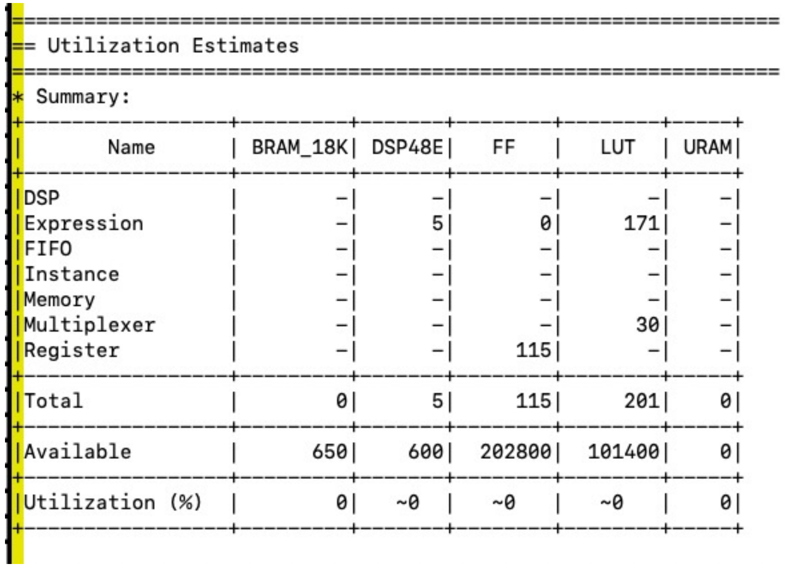


```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     156|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      30|    -|
|Register         |        -|      -|     147|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     147|     186|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|     ~0|      ~0|      ~0|    0|
+-----------------+---------+-------+--------+--------+-----+
```

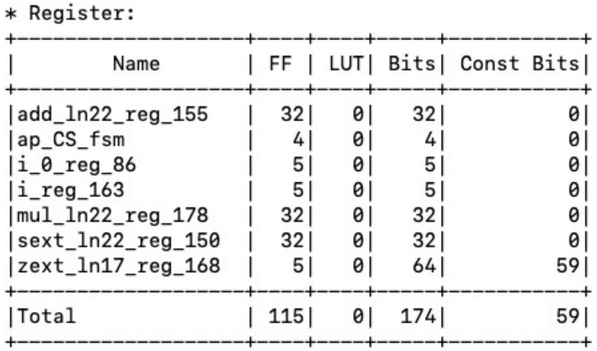


```
* Register:
+----------------------+----+----+-----+-----------+
|         Name         | FF | LUT| Bits| Const Bits|
+----------------------+----+----+-----+-----------+
|ap_CS_fsm             |   4|   0|    4|          0|
|i_0_reg_82            |   5|   0|    5|          0|
|i_reg_167             |   5|   0|    5|          0|
|mul_ln20_reg_182      |  32|   0|   32|          0|
|res_reg_154           |  32|   0|   32|          0|
|sext_ln20_1_reg_159   |  32|   0|   32|          0|
|sext_ln20_reg_149     |  32|   0|   32|          0|
|zext_ln15_reg_172     |   5|   0|   64|         59|
+----------------------+----+----+-----+-----------+
|Total                 | 147|   0|  206|         59|
+----------------------+----+----+-----+-----------+
```

WITH PRAGMA



```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     171|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      30|    -|
|Register         |        -|      -|     115|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     115|     201|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|     ~0|      ~0|      ~0|    0|
+-----------------+---------+-------+--------+--------+-----+
```



```
* Register:
+--------------------+----+----+-----+-----------+
|        Name        | FF | LUT| Bits| Const Bits|
+--------------------+----+----+-----+-----------+
|add_ln22_reg_155    |  32|   0|   32|          0|
|ap_CS_fsm           |   4|   0|    4|          0|
|i_0_reg_86          |   5|   0|    5|          0|
|i_reg_163           |   5|   0|    5|          0|
|mul_ln22_reg_178    |  32|   0|   32|          0|
|sext_ln22_reg_150   |  32|   0|   32|          0|
|zext_ln17_reg_168   |   5|   0|   64|         59|
+--------------------+----+----+-----+-----------+
|Total               | 115|   0|  174|         59|
+--------------------+----+----+-----+-----------+
```

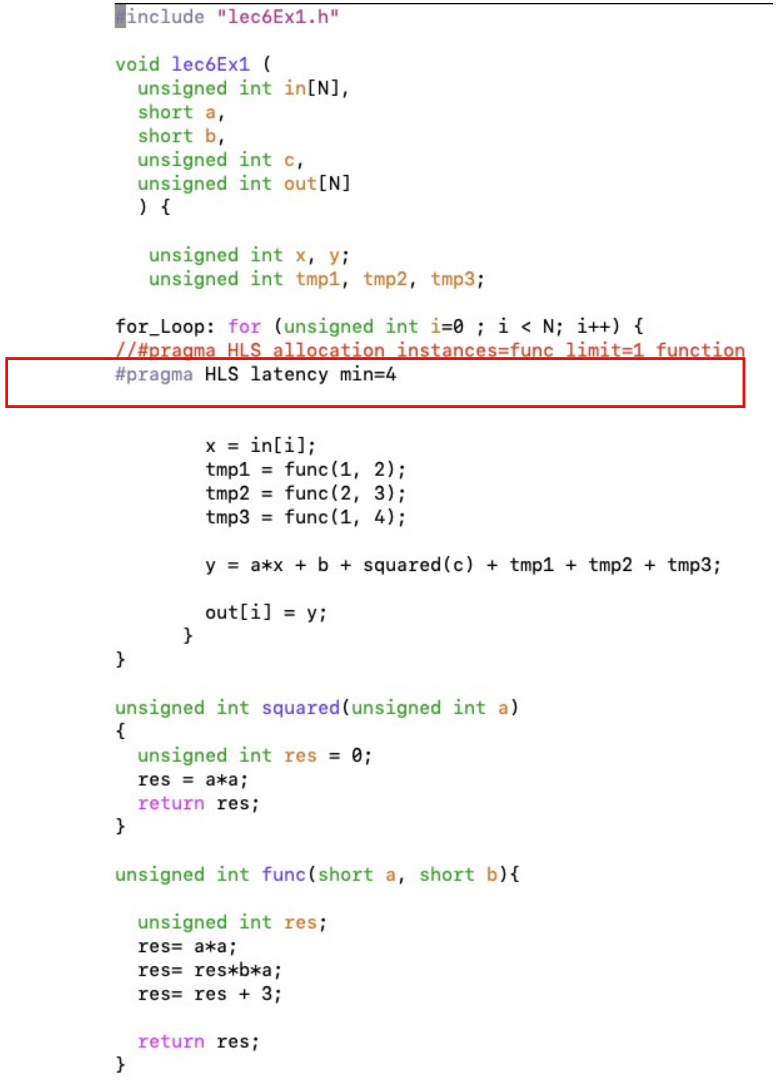
Pragma HLS Latency

#pragma HLS latency min=<int> max=<int>

- HLS always tries to minimize latency in the design

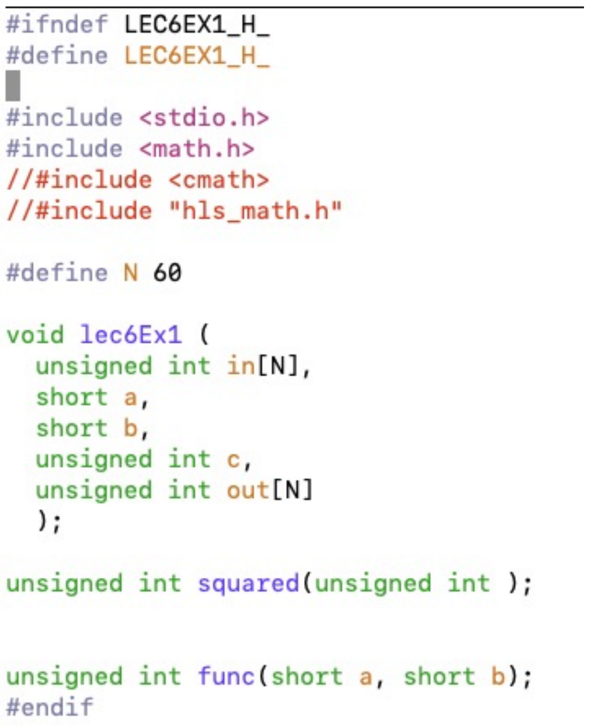
- When LATENCY pragma is specified
  - ***Min < Latency < Max***: Constraint is satisfied, No further optimization
  - ***Latency < min***: It extends latency to the specified value, potentially increasing sharing
  - ***Latency > max***: Increases effort to achieve the constraints
    - Still unsuccessful: issue a warning & produce design with the smallest achievable latency in excess of maximum

Pragma HLS Latency

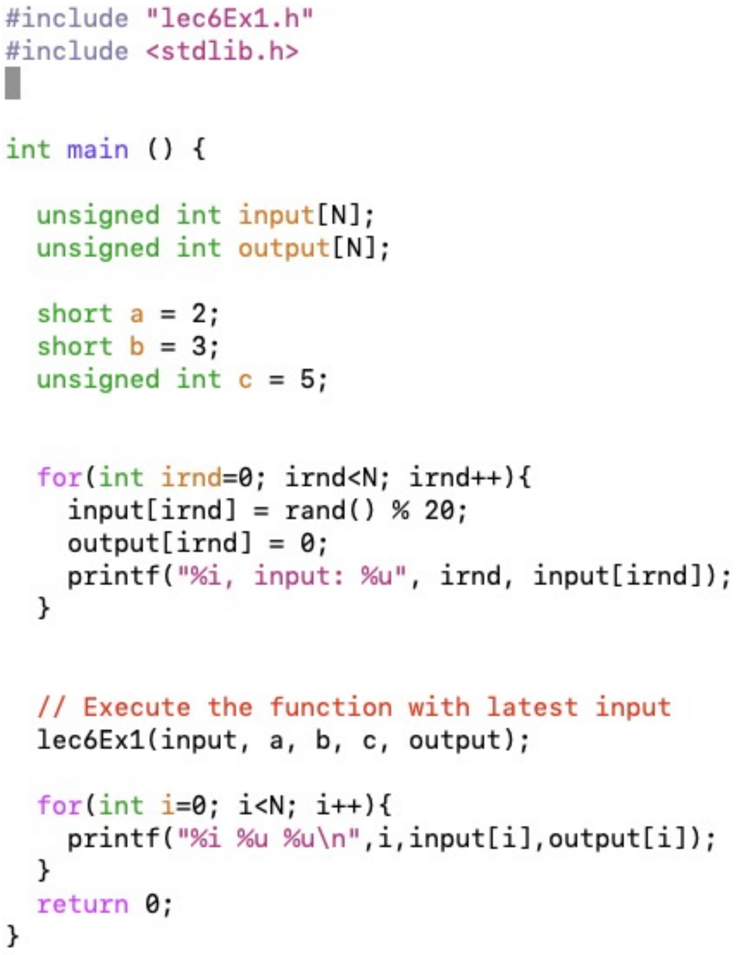


```
#include "lec6Ex1.h"

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  ) {

   unsigned int x, y;
   unsigned int tmp1, tmp2, tmp3;

for_Loop: for (unsigned int i=0 ; i < N; i++) {
//#pragma HLS allocation instances=func limit=1 function
#pragma HLS latency min=4


        x = in[i];
        tmp1 = func(1, 2);
        tmp2 = func(2, 3);
        tmp3 = func(1, 4);

        y = a*x + b + squared(c) + tmp1 + tmp2 + tmp3;

        out[i] = y;
      }
}

unsigned int squared(unsigned int a)
{
  unsigned int res = 0;
  res = a*a;
  return res;
}

unsigned int func(short a, short b){

  unsigned int res;
  res= a*a;
  res= res*b*a;
  res= res + 3;

  return res;
}
```

#pragma HLS latency min=4



```
#ifndef LEC6EX1_H_
#define LEC6EX1_H_

#include <stdio.h>
#include <math.h>
//#include <cmath>
//#include "hls_math.h"

#define N 60

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  );

unsigned int squared(unsigned int );


unsigned int func(short a, short b);
#endif
```

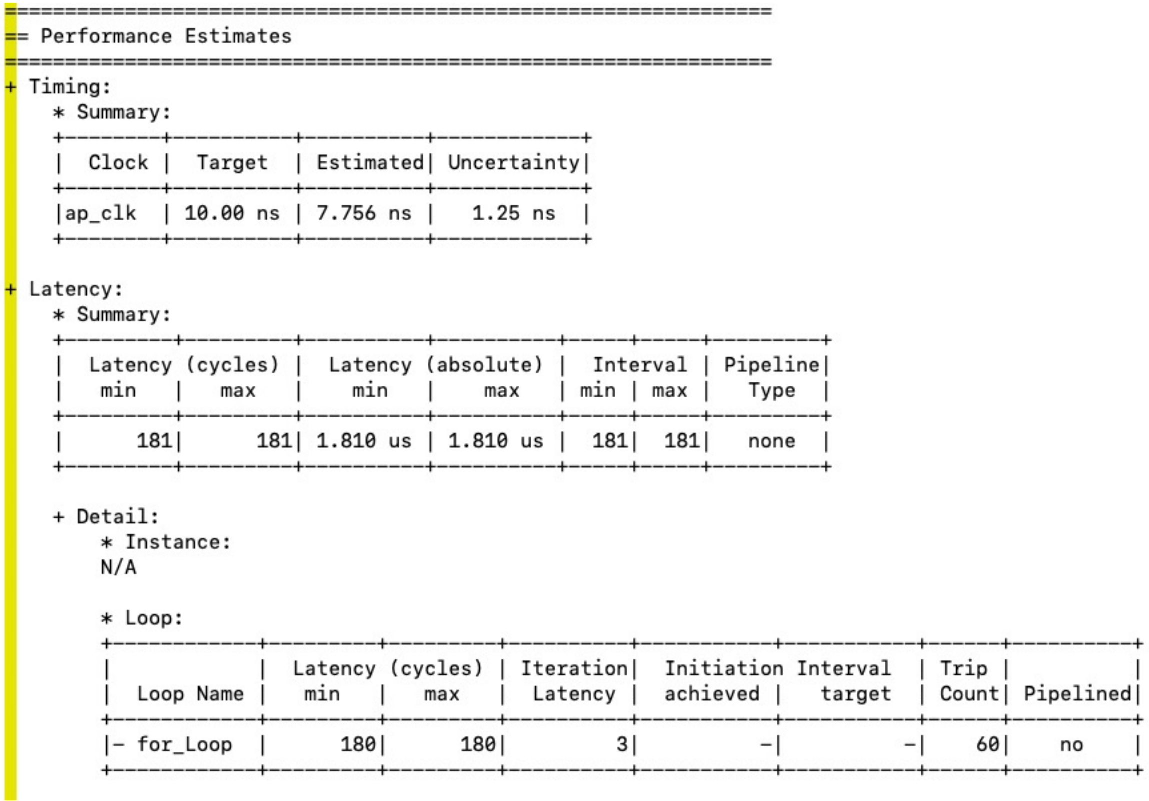


```
#include "lec6Ex1.h"
#include <stdlib.h>


int main () {

  unsigned int input[N];
  unsigned int output[N];

  short a = 2;
  short b = 3;
  unsigned int c = 5;


  for(int irnd=0; irnd<N; irnd++){
    input[irnd] = rand() % 20;
    output[irnd] = 0;
    printf("%i, input: %u", irnd, input[irnd]);
  }


  // Execute the function with latest input
  lec6Ex1(input, a, b, c, output);

  for(int i=0; i<N; i++){
    printf("%i %u %u\n",i,input[i],output[i]);
  }
  return 0;
}
```

Pragma HLS Latency

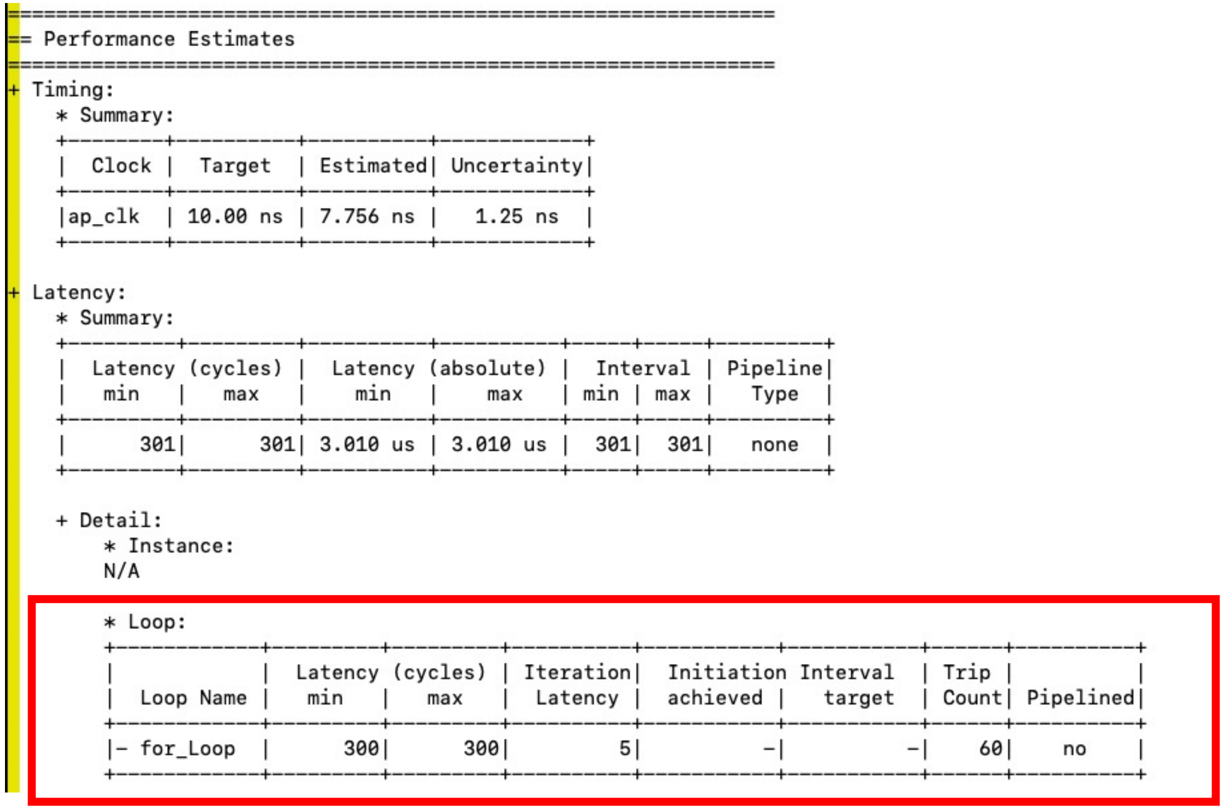
#pragma HLS latency min=4

WITHOUT HLS Latency PRAGMA



```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |      181|      181| 1.810 us | 1.810 us |  181|  181|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |- for_Loop  |      180|      180|         3|          -|          -|    60|    no    |
        +------------+---------+---------+----------+-----------+-----------+------+----------+
```

WITH HLS Latency PRAGMA

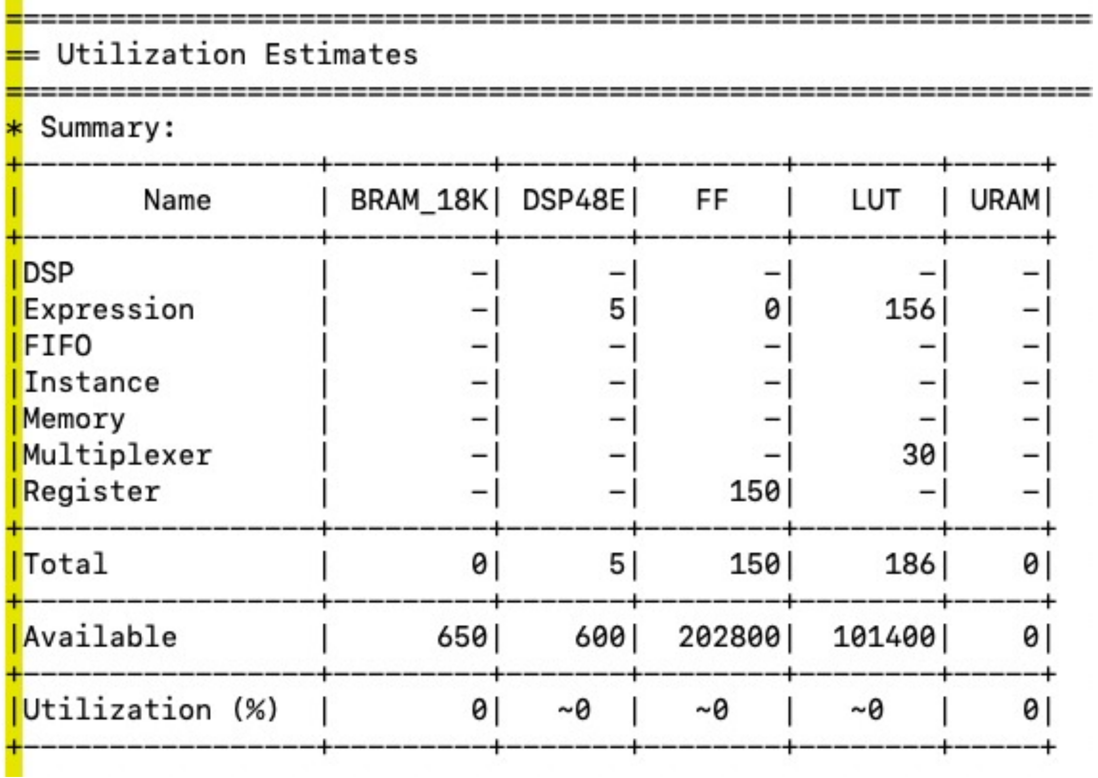


```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |      301|      301| 3.010 us | 3.010 us |  301|  301|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |- for_Loop  |      300|      300|         5|          -|          -|    60|    no    |
        +------------+---------+---------+----------+-----------+-----------+------+----------+
```

Pragma HLS Latency

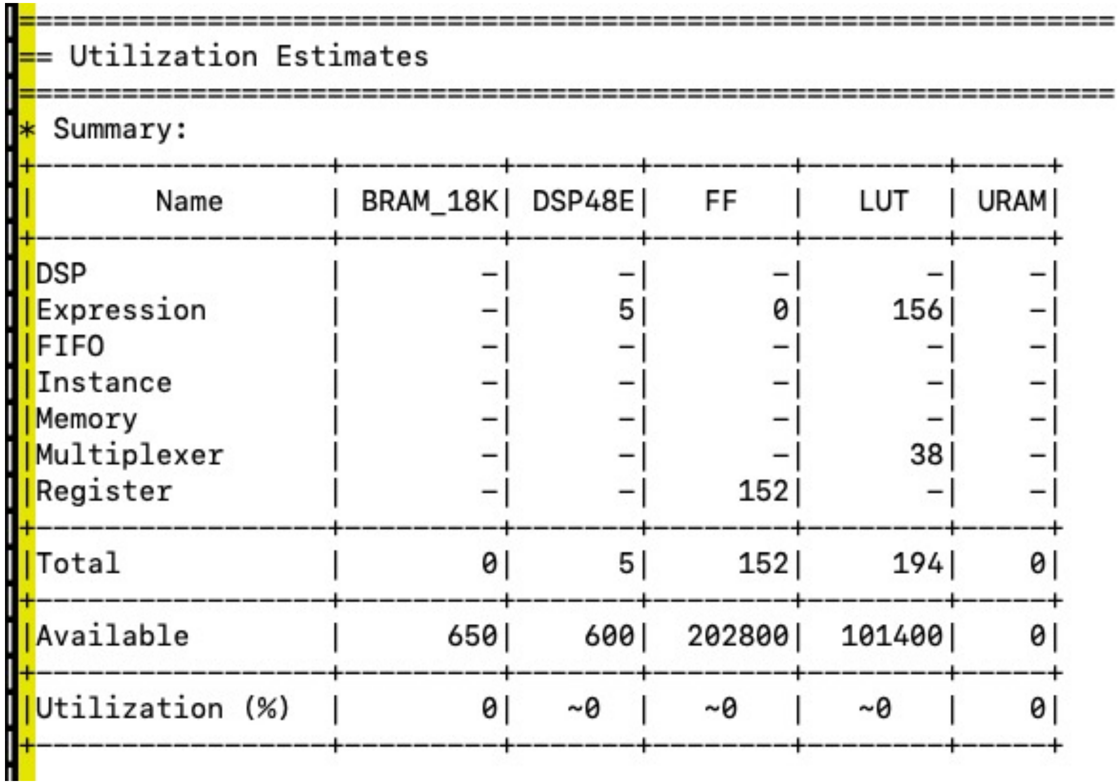
#pragma HLS latency min=4

WITHOUT HLS Latency PRAGMA



```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     156|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      30|    -|
|Register         |        -|      -|     150|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     150|     186|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|   ~0  |   ~0   |   ~0   |    0|
+-----------------+---------+-------+--------+--------+-----+
```

WITH HLS Latency PRAGMA



```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     156|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      38|    -|
|Register         |        -|      -|     152|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     152|     194|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|   ~0  |   ~0   |   ~0   |    0|
+-----------------+---------+-------+--------+--------+-----+
```

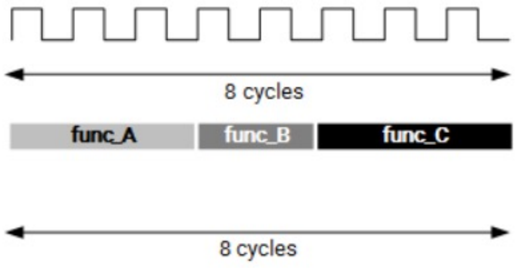
Not much change in the resources

Pragma HLS Dataflow

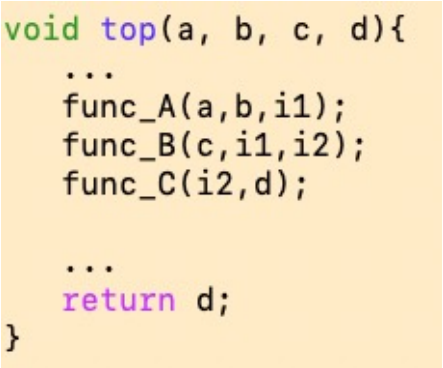
Task-level pipeline

#pragma HLS dataflow

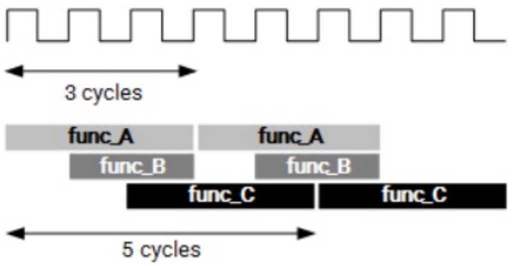
- Enables task-level pipelining: allow functions and loops to overlap in their operation
  - Increases the concurrency of the RTL implementation & thus the overall throughput of the design
- In the absence of any directives that limit resources (like pragma HLS allocation), HLS seeks to minimize latency & improve concurrency
  - Data dependencies can limit this, hence proper dataflow is needed



Without DATAFLOW pipelining



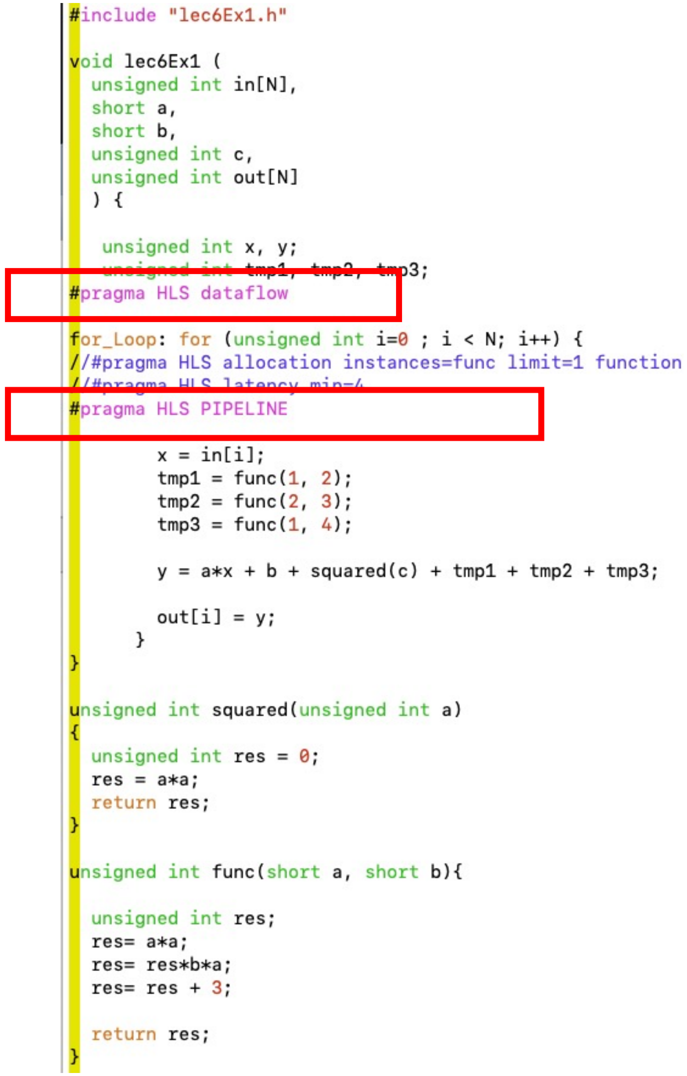
```
void top(a, b, c, d){
   ...
   func_A(a,b,i1);
   func_B(c,i1,i2);
   func_C(i2,d);

   ...
   return d;
}
```



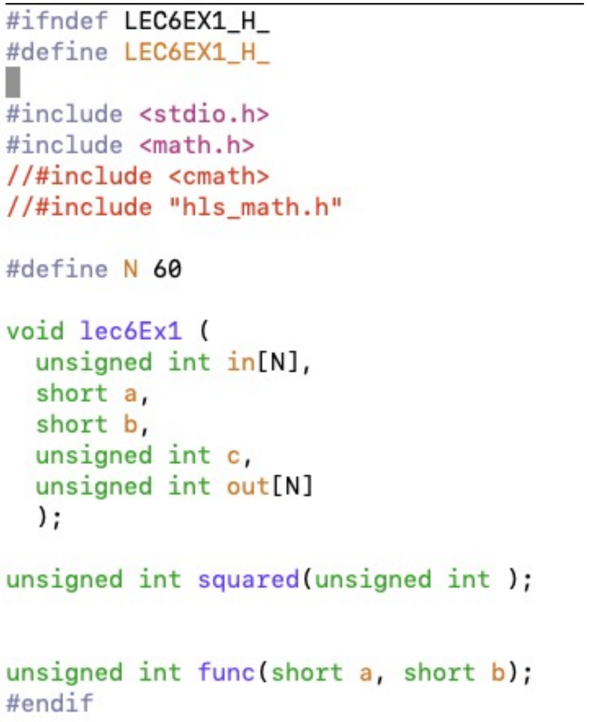
With DATAFLOW pipelining

Pragma HLS Dataflow

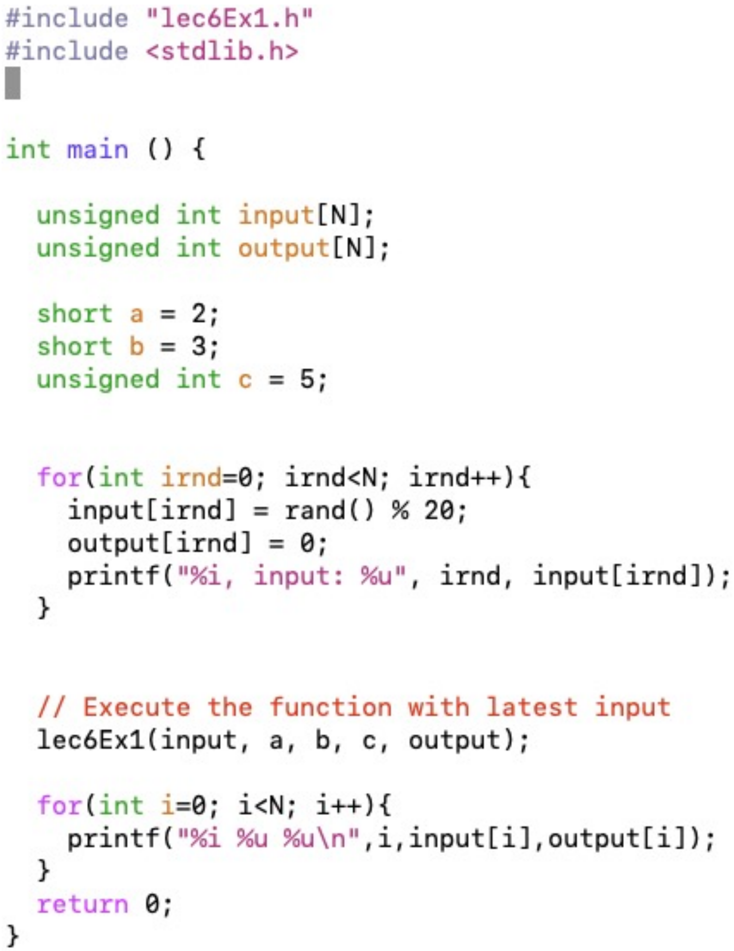
#pragma HLS dataflow



```
#include "lec6Ex1.h"

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  ) {

   unsigned int x, y;
   unsigned int tmp1, tmp2, tmp3;
#pragma HLS dataflow

for_Loop: for (unsigned int i=0 ; i < N; i++) {
//#pragma HLS allocation instances=func limit=1 function
//#pragma HLS latency min=4
#pragma HLS PIPELINE

        x = in[i];
        tmp1 = func(1, 2);
        tmp2 = func(2, 3);
        tmp3 = func(1, 4);

        y = a*x + b + squared(c) + tmp1 + tmp2 + tmp3;

        out[i] = y;
      }
}

unsigned int squared(unsigned int a)
{
  unsigned int res = 0;
  res = a*a;
  return res;
}

unsigned int func(short a, short b){

  unsigned int res;
  res= a*a;
  res= res*b*a;
  res= res + 3;

  return res;
}
```



```
#ifndef LEC6EX1_H_
#define LEC6EX1_H_

#include <stdio.h>
#include <math.h>
//#include <cmath>
//#include "hls_math.h"

#define N 60

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  );

unsigned int squared(unsigned int );


unsigned int func(short a, short b);
#endif
```

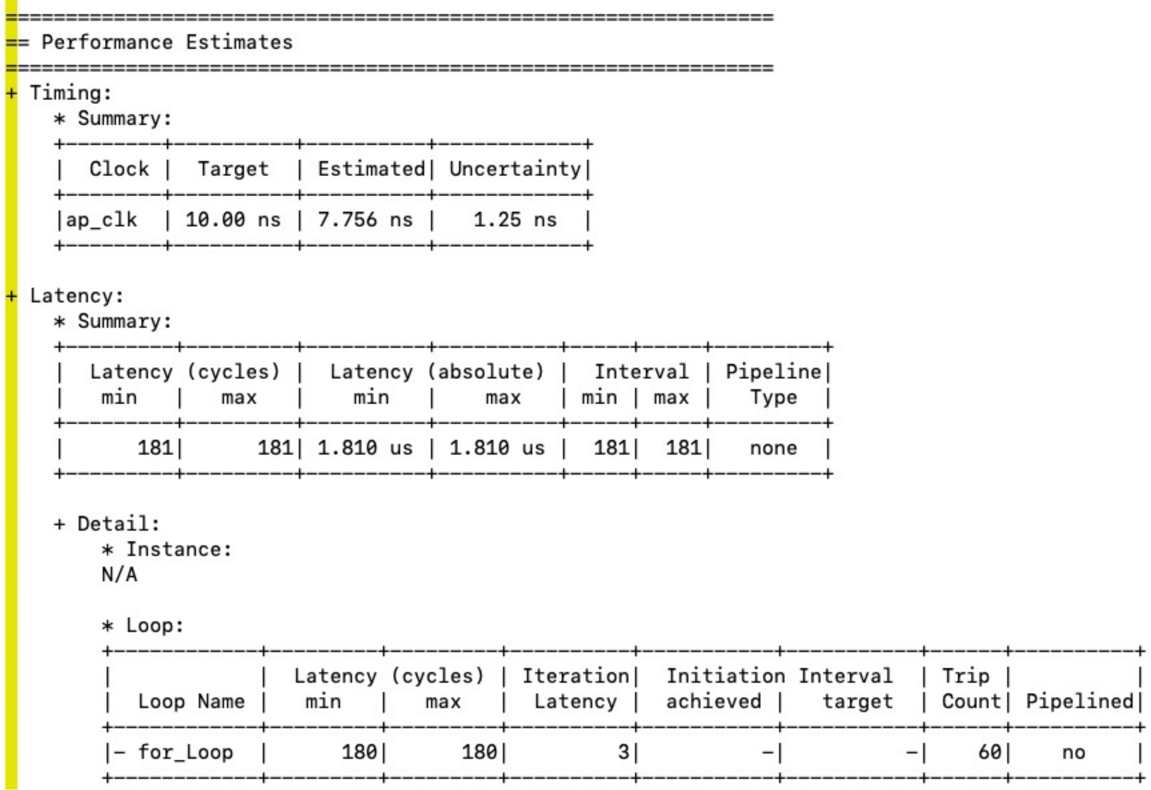


```
#include "lec6Ex1.h"
#include <stdlib.h>


int main () {

  unsigned int input[N];
  unsigned int output[N];

  short a = 2;
  short b = 3;
  unsigned int c = 5;


  for(int irnd=0; irnd<N; irnd++){
    input[irnd] = rand() % 20;
    output[irnd] = 0;
    printf("%i, input: %u", irnd, input[irnd]);
  }


  // Execute the function with latest input
  lec6Ex1(input, a, b, c, output);

  for(int i=0; i<N; i++){
    printf("%i %u %u\n",i,input[i],output[i]);
  }
  return 0;
}
```

Pragma HLS Dataflow

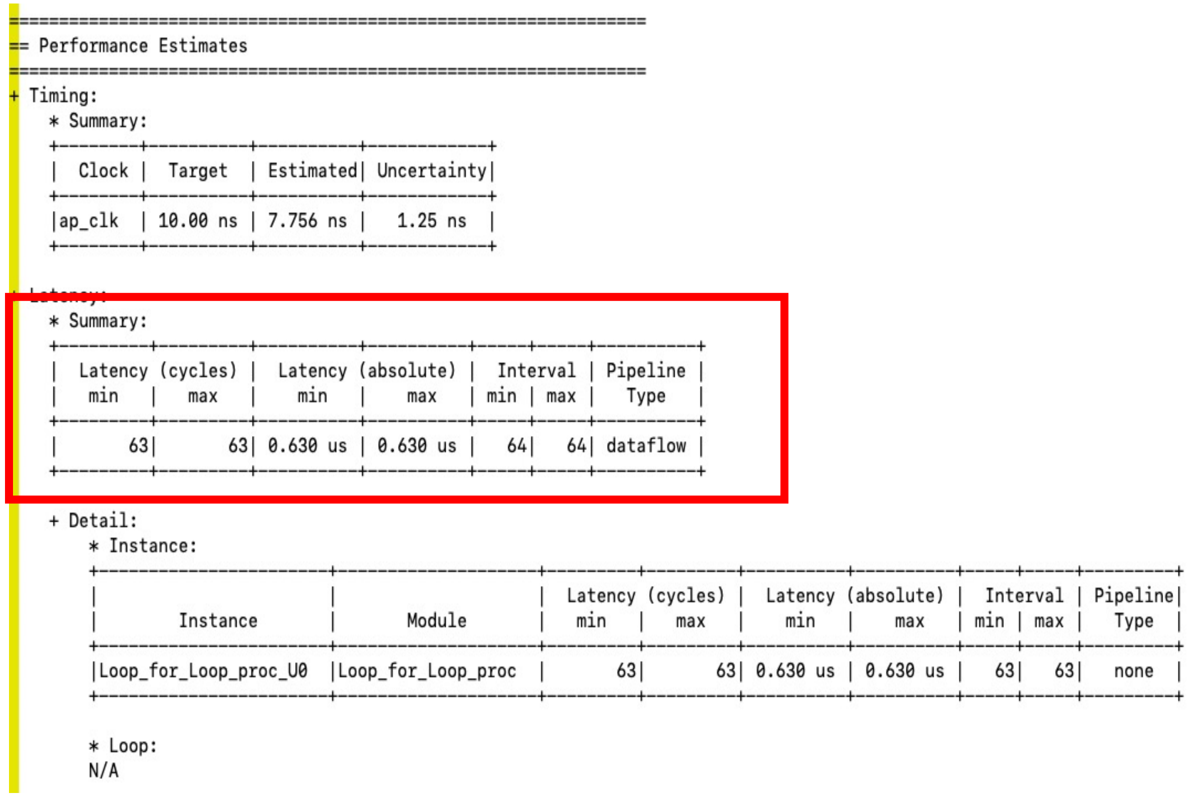
#pragma HLS dataflow

Without DATAFLOW pipelining



```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |      181|      181| 1.810 us | 1.810 us |  181|  181|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +-------------+---------+---------+----------+-----------+-----------+------+----------+
        |             |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name  |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +-------------+---------+---------+----------+-----------+-----------+------+----------+
        |- for_Loop   |      180|      180|         3|          -|          -|    60|    no    |
        +-------------+---------+---------+----------+-----------+-----------+------+----------+
```

With DATAFLOW pipelining

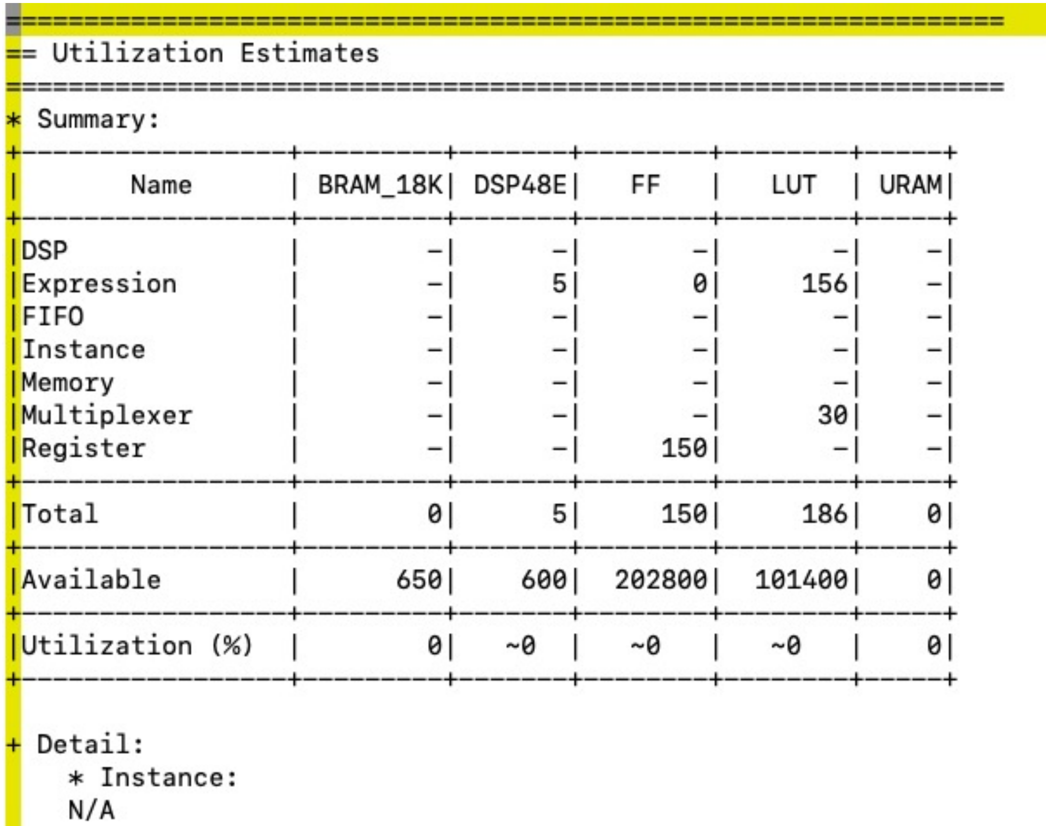


```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+----------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline |
    |   min   |   max   |    min   |    max   | min | max |   Type   |
    +---------+---------+----------+----------+-----+-----+----------+
    |       63|       63| 0.630 us | 0.630 us |   64|   64| dataflow |
    +---------+---------+----------+----------+-----+-----+----------+

    + Detail:
        * Instance:
        +-----------------------+---------------------+---------+---------+----------+----------+-----+-----+---------+
        |                       |                     |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
        |        Instance       |        Module       |   min   |   max   |    min   |    max   | min | max |   Type  |
        +-----------------------+---------------------+---------+---------+----------+----------+-----+-----+---------+
        |Loop_for_Loop_proc_U0  |Loop_for_Loop_proc   |       63|       63| 0.630 us | 0.630 us |   63|   63|   none  |
        +-----------------------+---------------------+---------+---------+----------+----------+-----+-----+---------+

        * Loop:
        N/A
```

Pragma HLS Dataflow

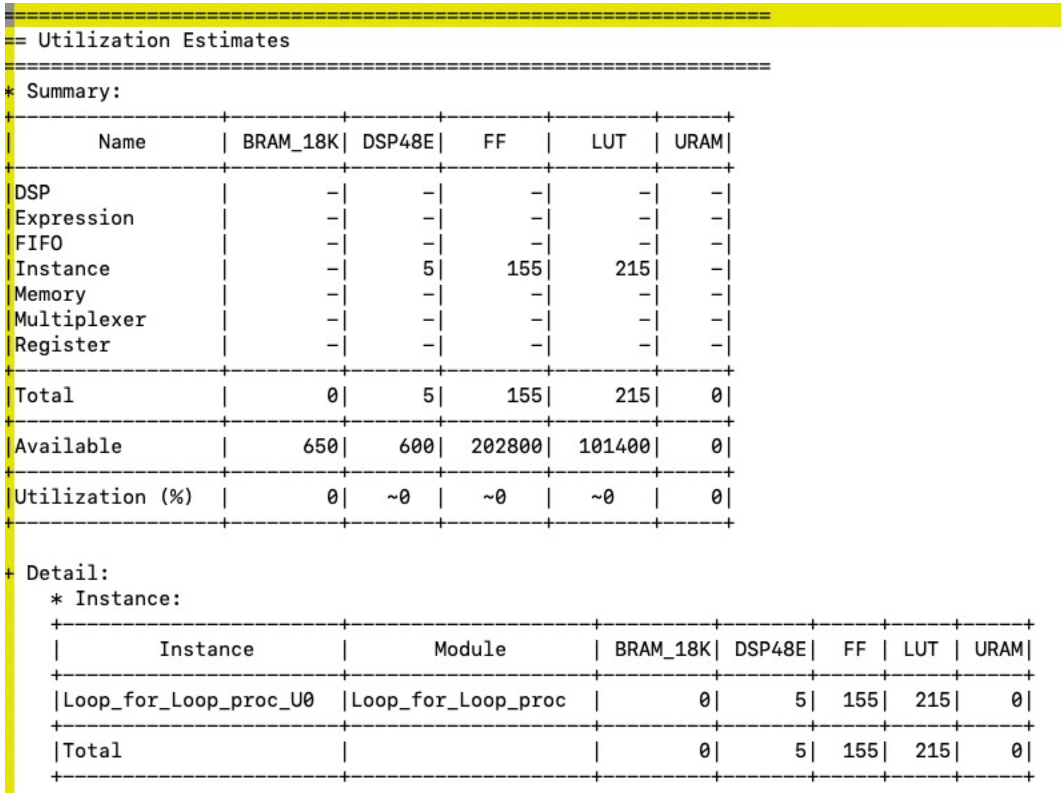
#pragma HLS dataflow

Without DATAFLOW pipelining



```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     156|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      30|    -|
|Register         |        -|      -|     150|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     150|     186|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|     ~0|      ~0|      ~0|    0|
+-----------------+---------+-------+--------+--------+-----+

+ Detail:
    * Instance:
    N/A
```

With DATAFLOW pipelining

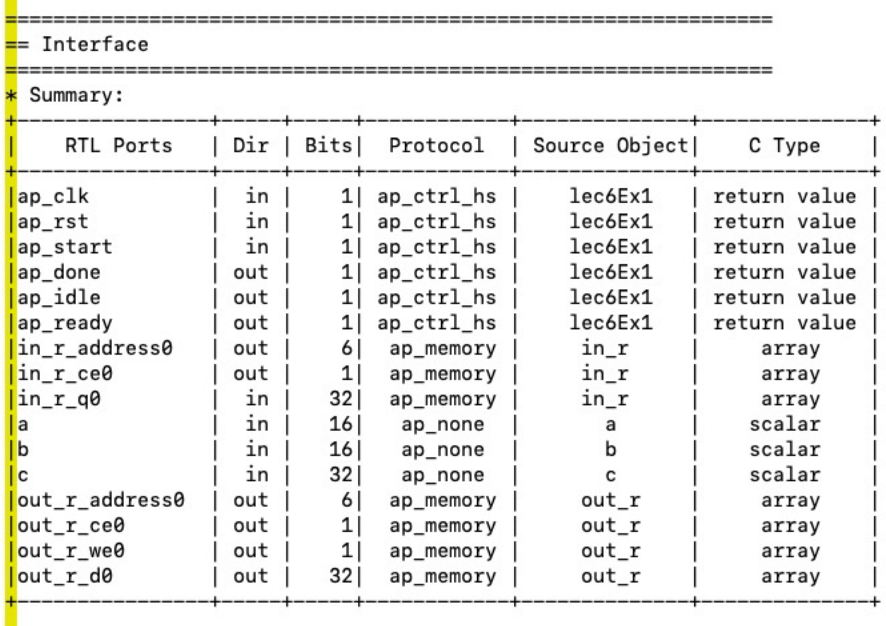


```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      -|       -|       -|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      5|     155|     215|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|       -|    -|
|Register         |        -|      -|       -|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     155|     215|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|     ~0|      ~0|      ~0|    0|
+-----------------+---------+-------+--------+--------+-----+

+ Detail:
    * Instance:
    +----------------------+-------------------+---------+-------+-----+-----+-----+
    |       Instance       |       Module      | BRAM_18K| DSP48E|  FF | LUT | URAM|
    +----------------------+-------------------+---------+-------+-----+-----+-----+
    |Loop_for_Loop_proc_U0 |Loop_for_Loop_proc |        0|      5|  155|  215|    0|
    +----------------------+-------------------+---------+-------+-----+-----+-----+
    |Total                 |                   |        0|      5|  155|  215|    0|
    +----------------------+-------------------+---------+-------+-----+-----+-----+
```

Pragma HLS Dataflow

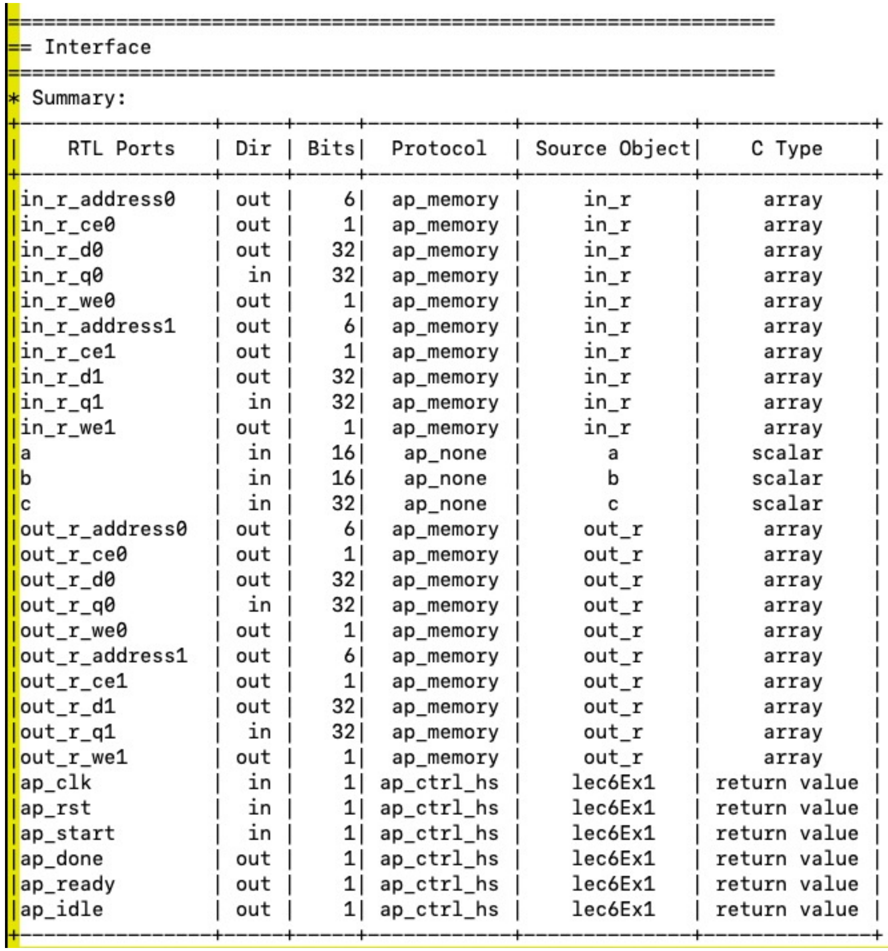
#pragma HLS dataflow

Without DATAFLOW pipelining



```
================================================================
== Interface
================================================================
* Summary:
+-----------------+-----+-----+------------+--------------+--------------+
|    RTL Ports    | Dir | Bits|  Protocol  | Source Object|    C Type    |
+-----------------+-----+-----+------------+--------------+--------------+
|ap_clk           |  in |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|ap_rst           |  in |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|ap_start         |  in |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|ap_done          | out |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|ap_idle          | out |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|ap_ready         | out |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|in_r_address0    | out |    6|  ap_memory |     in_r     |     array    |
|in_r_ce0         | out |    1|  ap_memory |     in_r     |     array    |
|in_r_q0          |  in |   32|  ap_memory |     in_r     |     array    |
|a                |  in |   16|   ap_none  |       a      |    scalar    |
|b                |  in |   16|   ap_none  |       b      |    scalar    |
|c                |  in |   32|   ap_none  |       c      |    scalar    |
|out_r_address0   | out |    6|  ap_memory |     out_r    |     array    |
|out_r_ce0        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_we0        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_d0         | out |   32|  ap_memory |     out_r    |     array    |
+-----------------+-----+-----+------------+--------------+--------------+
```

With DATAFLOW pipelining



```
================================================================
== Interface
================================================================
* Summary:
+-----------------+-----+-----+------------+--------------+--------------+
|    RTL Ports    | Dir | Bits|  Protocol  | Source Object|    C Type    |
+-----------------+-----+-----+------------+--------------+--------------+
|in_r_address0    | out |    6|  ap_memory |     in_r     |     array    |
|in_r_ce0         | out |    1|  ap_memory |     in_r     |     array    |
|in_r_d0          | out |   32|  ap_memory |     in_r     |     array    |
|in_r_q0          |  in |   32|  ap_memory |     in_r     |     array    |
|in_r_we0         | out |    1|  ap_memory |     in_r     |     array    |
|in_r_address1    | out |    6|  ap_memory |     in_r     |     array    |
|in_r_ce1         | out |    1|  ap_memory |     in_r     |     array    |
|in_r_d1          | out |   32|  ap_memory |     in_r     |     array    |
|in_r_q1          |  in |   32|  ap_memory |     in_r     |     array    |
|in_r_we1         | out |    1|  ap_memory |     in_r     |     array    |
|a                |  in |   16|   ap_none  |       a      |    scalar    |
|b                |  in |   16|   ap_none  |       b      |    scalar    |
|c                |  in |   32|   ap_none  |       c      |    scalar    |
|out_r_address0   | out |    6|  ap_memory |     out_r    |     array    |
|out_r_ce0        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_d0         | out |   32|  ap_memory |     out_r    |     array    |
|out_r_q0         |  in |   32|  ap_memory |     out_r    |     array    |
|out_r_we0        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_address1   | out |    6|  ap_memory |     out_r    |     array    |
|out_r_ce1        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_d1         | out |   32|  ap_memory |     out_r    |     array    |
|out_r_q1         |  in |   32|  ap_memory |     out_r    |     array    |
|out_r_we1        | out |    1|  ap_memory |     out_r    |     array    |
|ap_clk           |  in |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|ap_rst           |  in |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|ap_start         |  in |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|ap_done          | out |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|ap_ready         | out |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
|ap_idle          | out |    1|  ap_ctrl_hs|    lec6Ex1   | return value |
+-----------------+-----+-----+------------+--------------+--------------+
```

Pragma HLS Inline

Function inlining

```
#pragma HLS inline <region | recursive | off>
```

- Removes a function as a separate entity in the hierarchy
- The function is dissolved into the calling function and no longer appears as a separate level of hierarchy in RTL design
- May improve area by allowing the components within the function to be better shared or optimized with the logic in the calling function

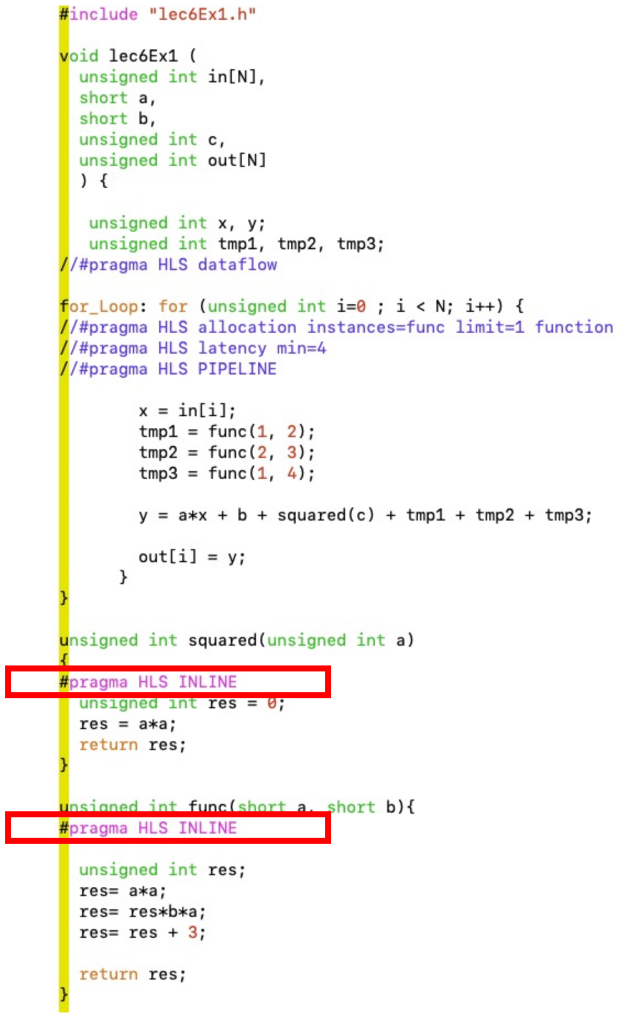
- ***Region***: Optionally, all functions (sub-functions) in the specified region are to be inlined

- ***Recursive***: Inlines all functions recursively within the specified function or region
  - By default, only one level of function inlining is performed

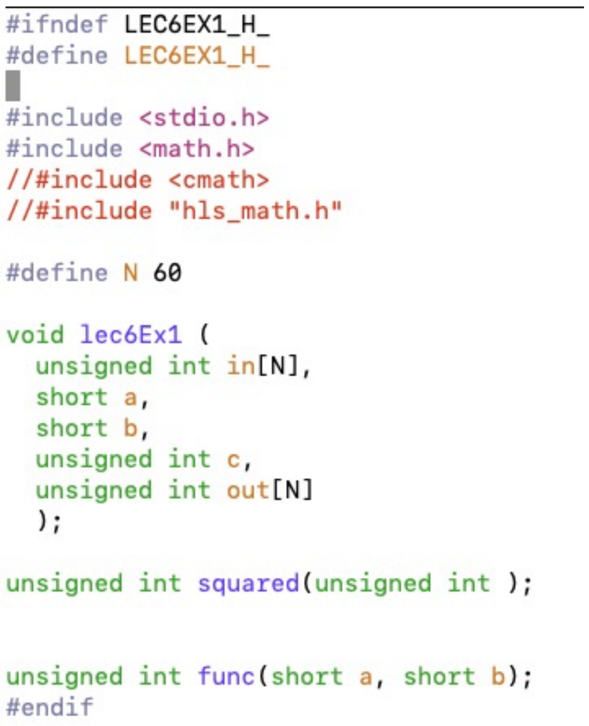
- ***Off:*** Disables function inlining to prevent specified functions from being inlined
  - For example, HLS automatically inlines small functions & with the off option, automatic inlining can be prevented

Pragma HLS Inline

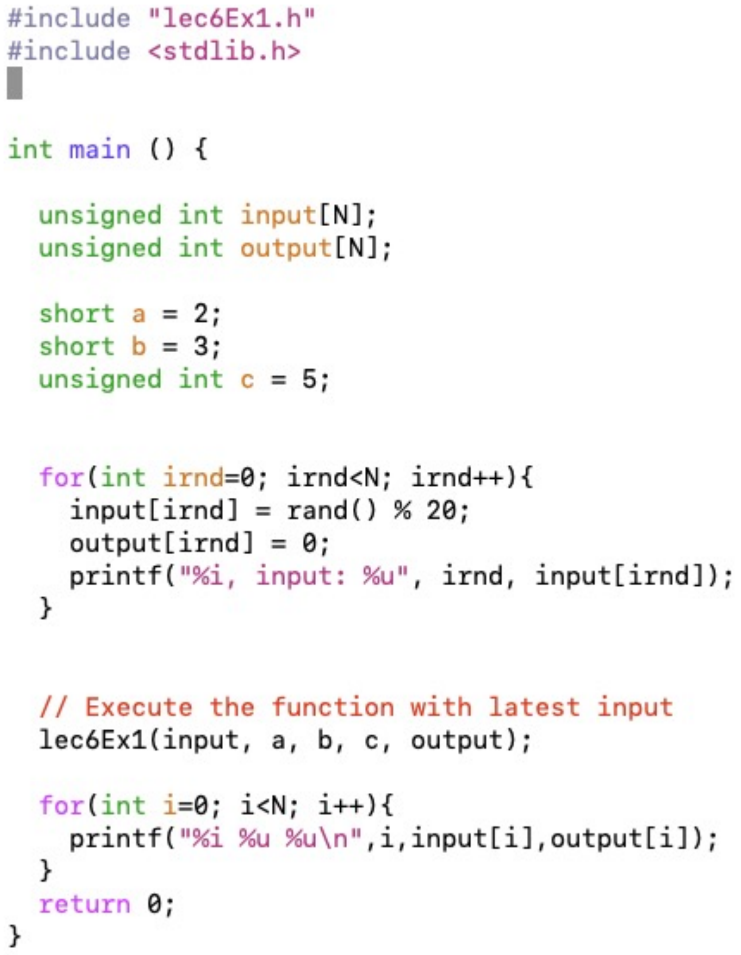
#pragma HLS inline



```
#include "lec6Ex1.h"

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  ) {

   unsigned int x, y;
   unsigned int tmp1, tmp2, tmp3;
//#pragma HLS dataflow

for_Loop: for (unsigned int i=0 ; i < N; i++) {
//#pragma HLS allocation instances=func limit=1 function
//#pragma HLS latency min=4
//#pragma HLS PIPELINE

        x = in[i];
        tmp1 = func(1, 2);
        tmp2 = func(2, 3);
        tmp3 = func(1, 4);

        y = a*x + b + squared(c) + tmp1 + tmp2 + tmp3;

        out[i] = y;
      }
}

unsigned int squared(unsigned int a)
{
#pragma HLS INLINE
  unsigned int res = 0;
  res = a*a;
  return res;
}

unsigned int func(short a, short b){
#pragma HLS INLINE

  unsigned int res;
  res= a*a;
  res= res*b*a;
  res= res + 3;

  return res;
}
```



```
#ifndef LEC6EX1_H_
#define LEC6EX1_H_

#include <stdio.h>
#include <math.h>
//#include <cmath>
//#include "hls_math.h"

#define N 60

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  );

unsigned int squared(unsigned int );


unsigned int func(short a, short b);
#endif
```

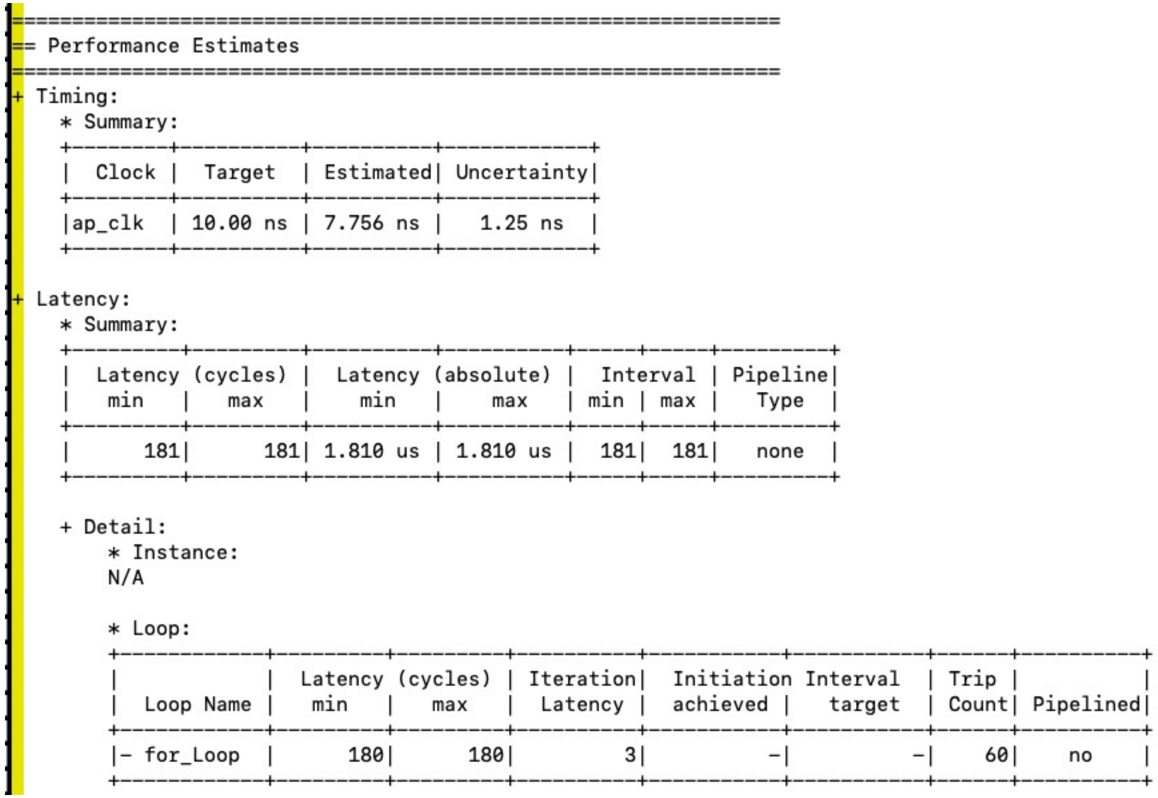


```
#include "lec6Ex1.h"
#include <stdlib.h>


int main () {

  unsigned int input[N];
  unsigned int output[N];

  short a = 2;
  short b = 3;
  unsigned int c = 5;


  for(int irnd=0; irnd<N; irnd++){
    input[irnd] = rand() % 20;
    output[irnd] = 0;
    printf("%i, input: %u", irnd, input[irnd]);
  }


  // Execute the function with latest input
  lec6Ex1(input, a, b, c, output);

  for(int i=0; i<N; i++){
    printf("%i %u %u\n",i,input[i],output[i]);
  }
  return 0;
}
```

Pragma HLS Inline

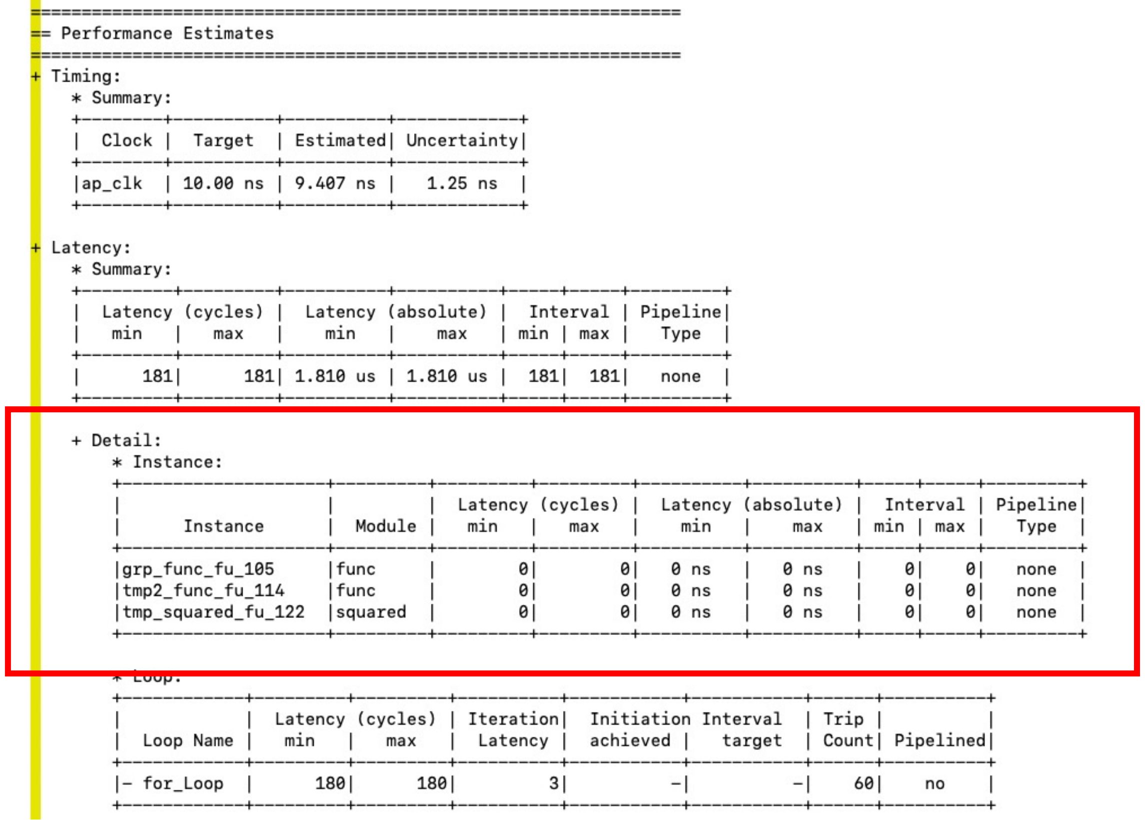
#pragma HLS inline **off**

With HLS INLINE



```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |      181|      181| 1.810 us | 1.810 us |  181|  181|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |- for_Loop  |      180|      180|         3|          -|          -|    60|    no    |
        +------------+---------+---------+----------+-----------+-----------+------+----------+
```

With HLS INLINE OFF

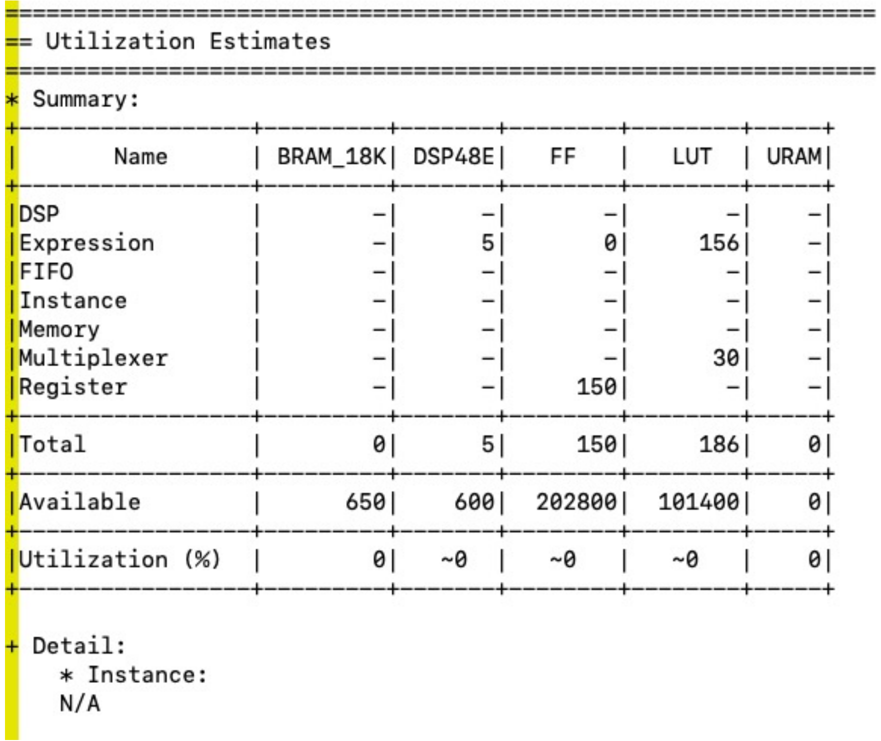


```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 9.407 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |      181|      181| 1.810 us | 1.810 us |  181|  181|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        +--------------------+----------+---------+---------+----------+----------+-----+-----+---------+
        |                    |          |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
        |      Instance      |  Module  |   min   |   max   |    min   |    max   | min | max |   Type  |
        +--------------------+----------+---------+---------+----------+----------+-----+-----+---------+
        |grp_func_fu_105     |func      |        0|        0|  0 ns    |  0 ns    |    0|    0|   none  |
        |tmp2_func_fu_114    |func      |        0|        0|  0 ns    |  0 ns    |    0|    0|   none  |
        |tmp_squared_fu_122  |squared   |        0|        0|  0 ns    |  0 ns    |    0|    0|   none  |
        +--------------------+----------+---------+---------+----------+----------+-----+-----+---------+

        * Loop:
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |- for_Loop  |      180|      180|         3|          -|          -|    60|    no    |
        +------------+---------+---------+----------+-----------+-----------+------+----------+
```

Pragma HLS Inline

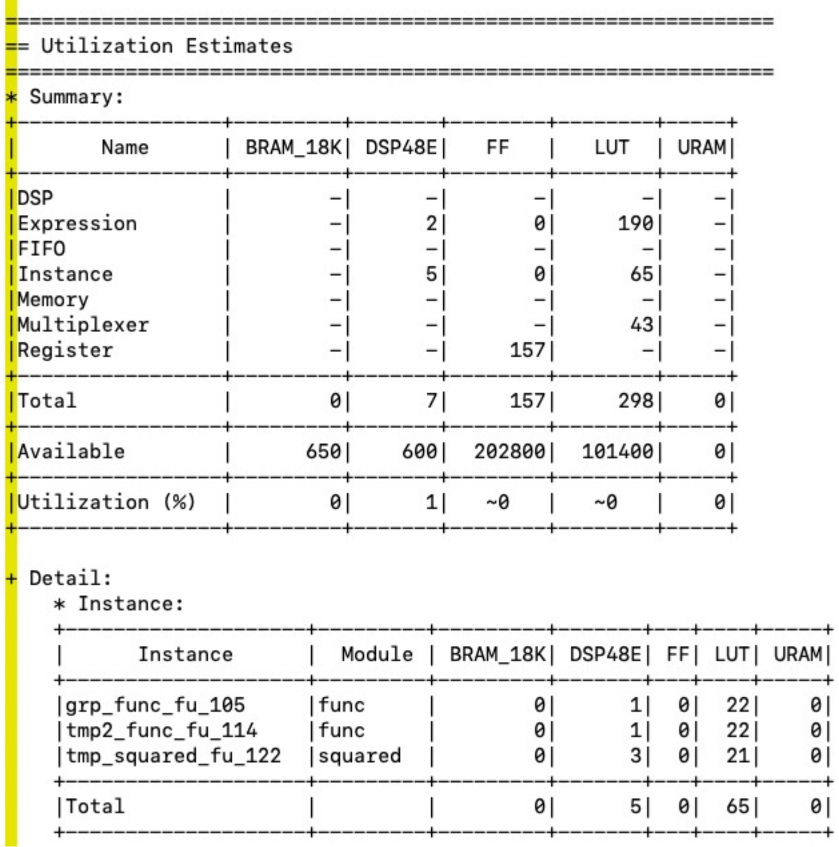
#pragma HLS inline **off**

With HLS INLINE



```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     156|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      30|    -|
|Register         |        -|      -|     150|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     150|     186|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|   ~0  |   ~0   |   ~0   |    0|
+-----------------+---------+-------+--------+--------+-----+

+ Detail:
    * Instance:
    N/A
```

With HLS INLINE OFF

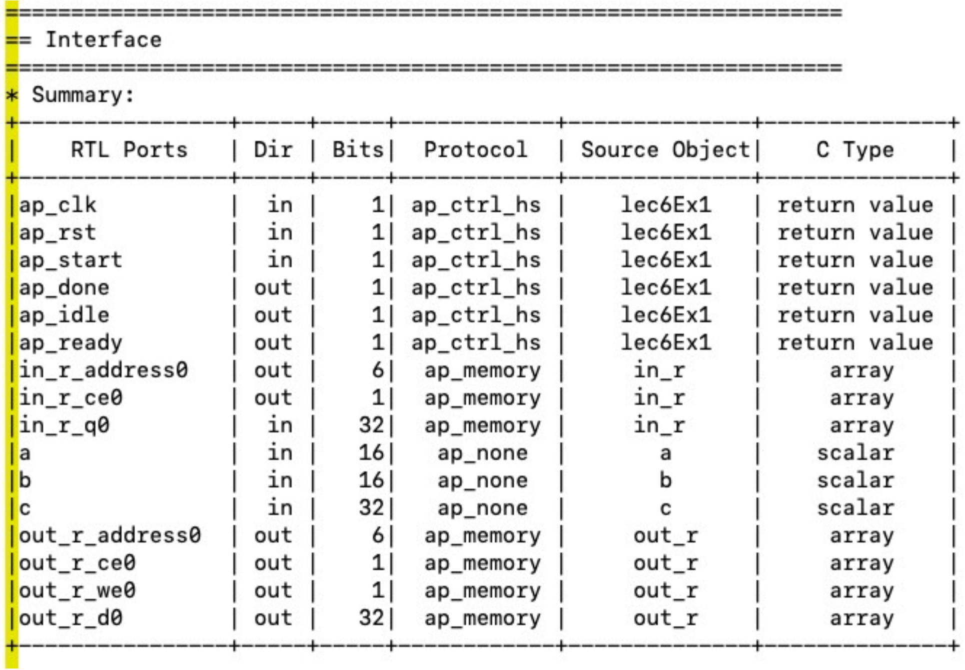


```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      2|       0|     190|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      5|       0|      65|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      43|    -|
|Register         |        -|      -|     157|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      7|     157|     298|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|      1|   ~0   |   ~0   |    0|
+-----------------+---------+-------+--------+--------+-----+

+ Detail:
    * Instance:
    +---------------------+----------+---------+-------+---+----+-----+
    |       Instance      |  Module  | BRAM_18K| DSP48E| FF| LUT| URAM|
    +---------------------+----------+---------+-------+---+----+-----+
    |grp_func_fu_105      |func      |        0|      1|  0|  22|    0|
    |tmp2_func_fu_114     |func      |        0|      1|  0|  22|    0|
    |tmp_squared_fu_122   |squared   |        0|      3|  0|  21|    0|
    +---------------------+----------+---------+-------+---+----+-----+
    |Total                |          |        0|      5|  0|  65|    0|
    +---------------------+----------+---------+-------+---+----+-----+
```

Pragma HLS Inline

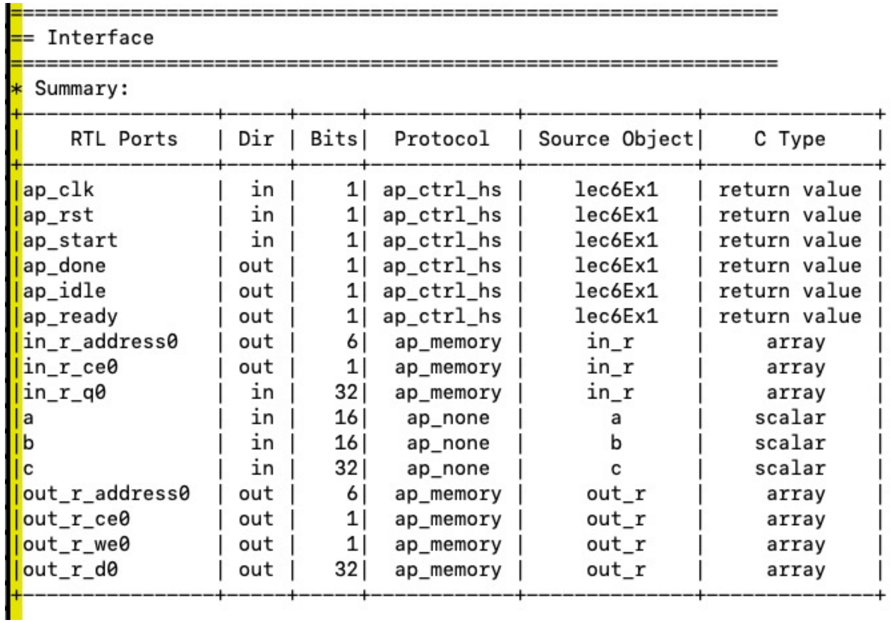
#pragma HLS inline **off**

With HLS INLINE



```
================================================================
== Interface
================================================================
* Summary:
+-----------------+-----+-----+------------+--------------+--------------+
|    RTL Ports    | Dir | Bits|  Protocol  | Source Object|    C Type    |
+-----------------+-----+-----+------------+--------------+--------------+
|ap_clk           |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_rst           |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_start         |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_done          | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_idle          | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_ready         | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|in_r_address0    | out |    6|  ap_memory |     in_r     |     array    |
|in_r_ce0         | out |    1|  ap_memory |     in_r     |     array    |
|in_r_q0          |  in |   32|  ap_memory |     in_r     |     array    |
|a                |  in |   16|   ap_none  |       a      |    scalar    |
|b                |  in |   16|   ap_none  |       b      |    scalar    |
|c                |  in |   32|   ap_none  |       c      |    scalar    |
|out_r_address0   | out |    6|  ap_memory |     out_r    |     array    |
|out_r_ce0        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_we0        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_d0         | out |   32|  ap_memory |     out_r    |     array    |
+-----------------+-----+-----+------------+--------------+--------------+
```

With HLS INLINE OFF



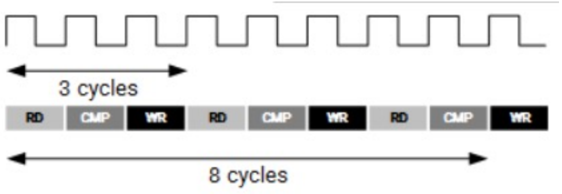
```
================================================================
== Interface
================================================================
* Summary:
+-----------------+-----+-----+------------+--------------+--------------+
|    RTL Ports    | Dir | Bits|  Protocol  | Source Object|    C Type    |
+-----------------+-----+-----+------------+--------------+--------------+
|ap_clk           |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_rst           |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_start         |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_done          | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_idle          | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_ready         | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|in_r_address0    | out |    6|  ap_memory |     in_r     |     array    |
|in_r_ce0         | out |    1|  ap_memory |     in_r     |     array    |
|in_r_q0          |  in |   32|  ap_memory |     in_r     |     array    |
|a                |  in |   16|   ap_none  |       a      |    scalar    |
|b                |  in |   16|   ap_none  |       b      |    scalar    |
|c                |  in |   32|   ap_none  |       c      |    scalar    |
|out_r_address0   | out |    6|  ap_memory |     out_r    |     array    |
|out_r_ce0        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_we0        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_d0         | out |   32|  ap_memory |     out_r    |     array    |
+-----------------+-----+-----+------------+--------------+--------------+
```

No change in the execution order

Pragma HLS Pipeline

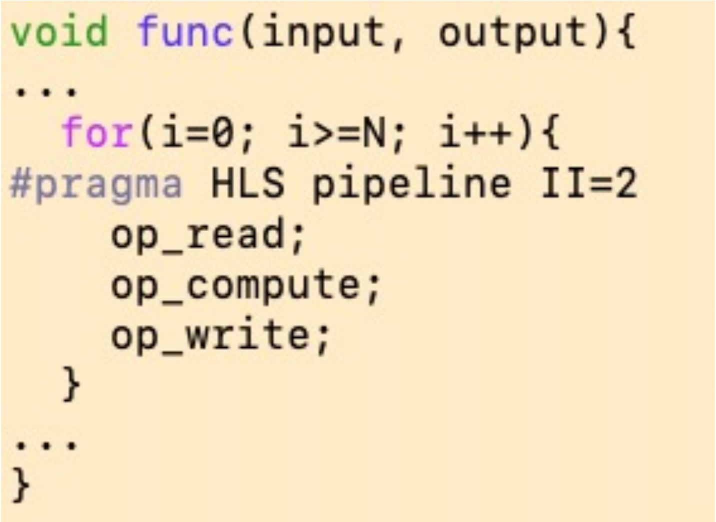
#pragma HLS pipeline II=<int>

- The PIPELINE pragma reduces the II for a function or loop by allowing the concurrent execution of operations
- A pipelined function or loop can process new inputs every <N> clock cycles
- If HLS can't create a design with the specified II, it issues a warning and creates a design with the lowest possible II

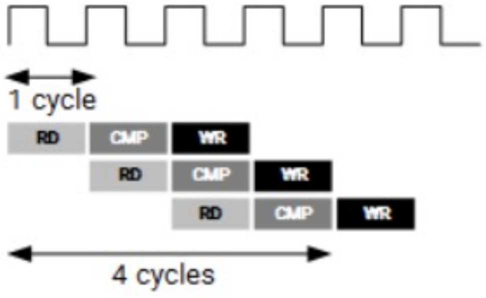


(A) Without Loop Pipelining

Without Loop pipelining



```
void func(input, output){
...
  for(i=0; i>=N; i++){
#pragma HLS pipeline II=2
      op_read;
      op_compute;
      op_write;
  }
...
}
```



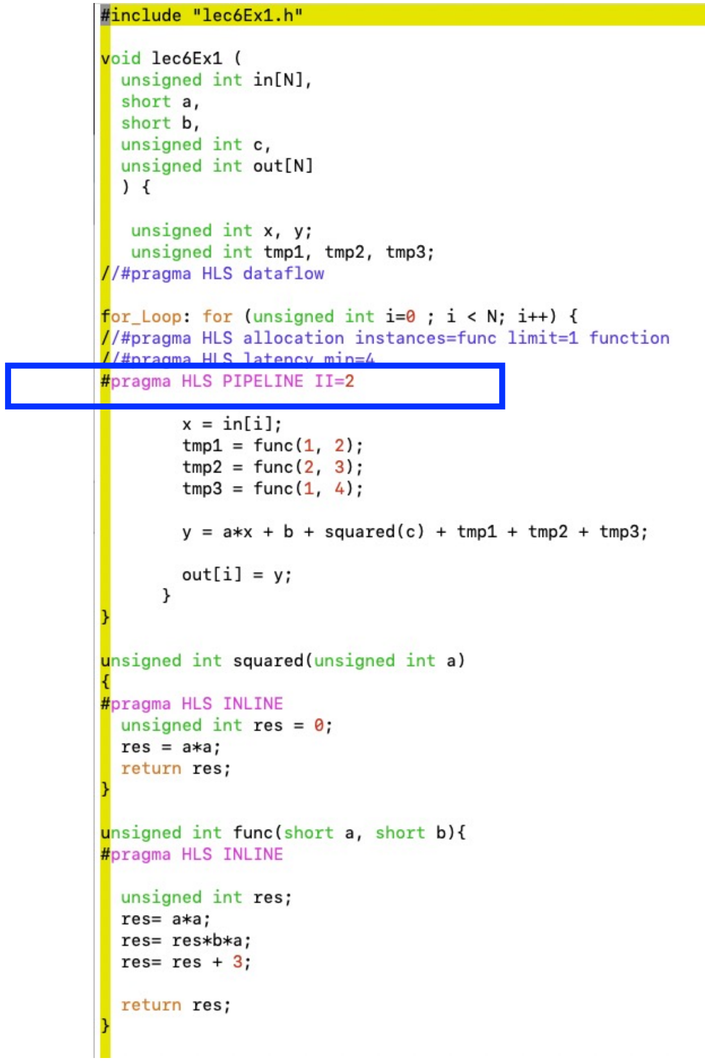
With Loop pipelining

Pragma HLS Pipeline

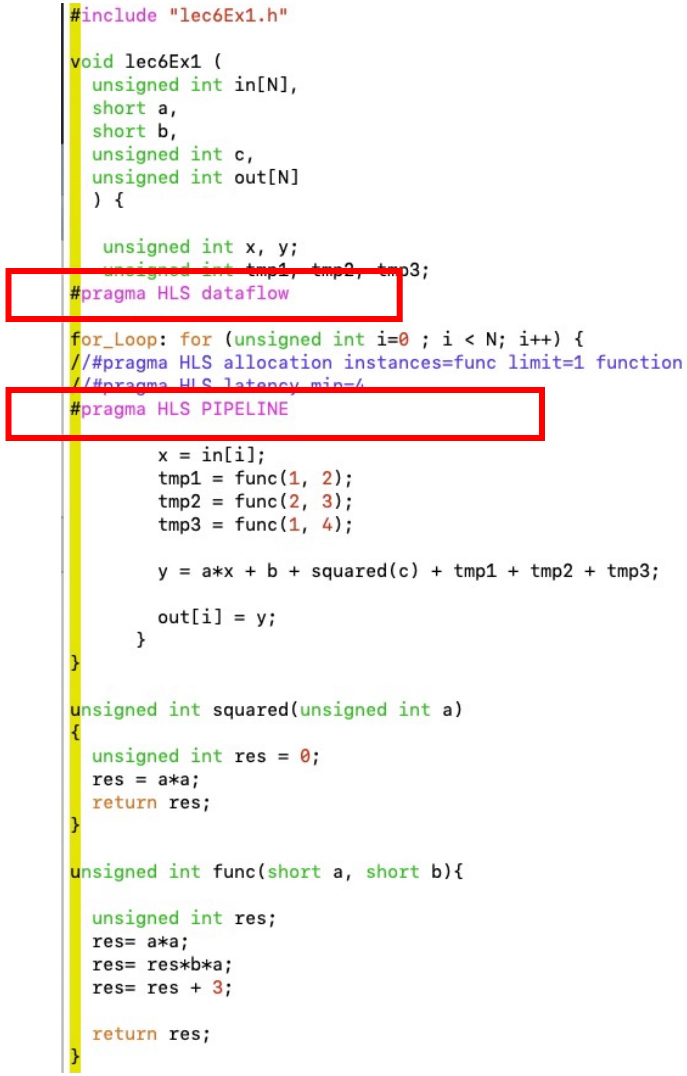
With HLS PIPELINE II=2

#pragma HLS pipeline II=2

With DATAFLOW pipelining



```
#include "lec6Ex1.h"

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  ) {

   unsigned int x, y;
   unsigned int tmp1, tmp2, tmp3;
//#pragma HLS dataflow

for_Loop: for (unsigned int i=0 ; i < N; i++) {
//#pragma HLS allocation instances=func limit=1 function
//#pragma HLS latency min=4
#pragma HLS PIPELINE II=2

        x = in[i];
        tmp1 = func(1, 2);
        tmp2 = func(2, 3);
        tmp3 = func(1, 4);

        y = a*x + b + squared(c) + tmp1 + tmp2 + tmp3;

        out[i] = y;
      }
}

unsigned int squared(unsigned int a)
{
#pragma HLS INLINE
  unsigned int res = 0;
  res = a*a;
  return res;
}

unsigned int func(short a, short b){
#pragma HLS INLINE

  unsigned int res;
  res= a*a;
  res= res*b*a;
  res= res + 3;

  return res;
}
```

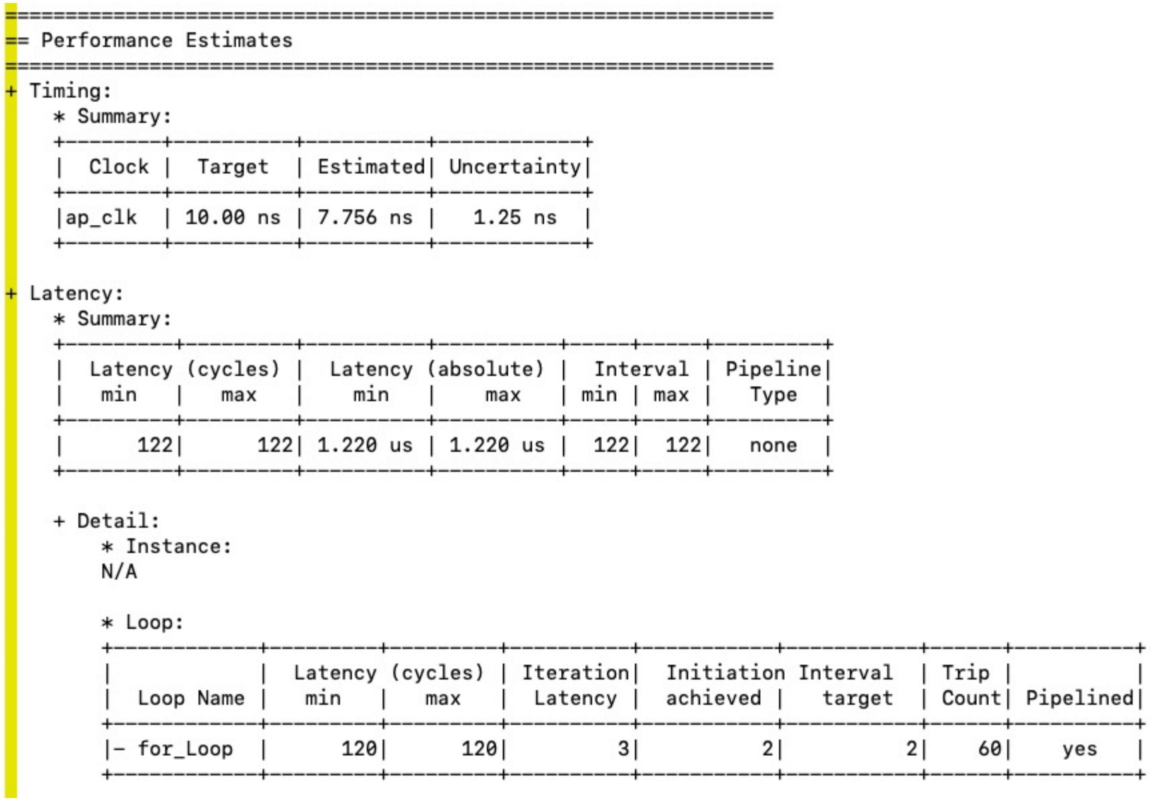


```
#include "lec6Ex1.h"

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  ) {

   unsigned int x, y;
   unsigned int tmp1, tmp2, tmp3;
#pragma HLS dataflow

for_Loop: for (unsigned int i=0 ; i < N; i++) {
//#pragma HLS allocation instances=func limit=1 function
//#pragma HLS latency min=4
#pragma HLS PIPELINE

        x = in[i];
        tmp1 = func(1, 2);
        tmp2 = func(2, 3);
        tmp3 = func(1, 4);

        y = a*x + b + squared(c) + tmp1 + tmp2 + tmp3;

        out[i] = y;
      }
}

unsigned int squared(unsigned int a)
{
  unsigned int res = 0;
  res = a*a;
  return res;
}

unsigned int func(short a, short b){

  unsigned int res;
  res= a*a;
  res= res*b*a;
  res= res + 3;

  return res;
}
```

Pragma HLS Pipeline

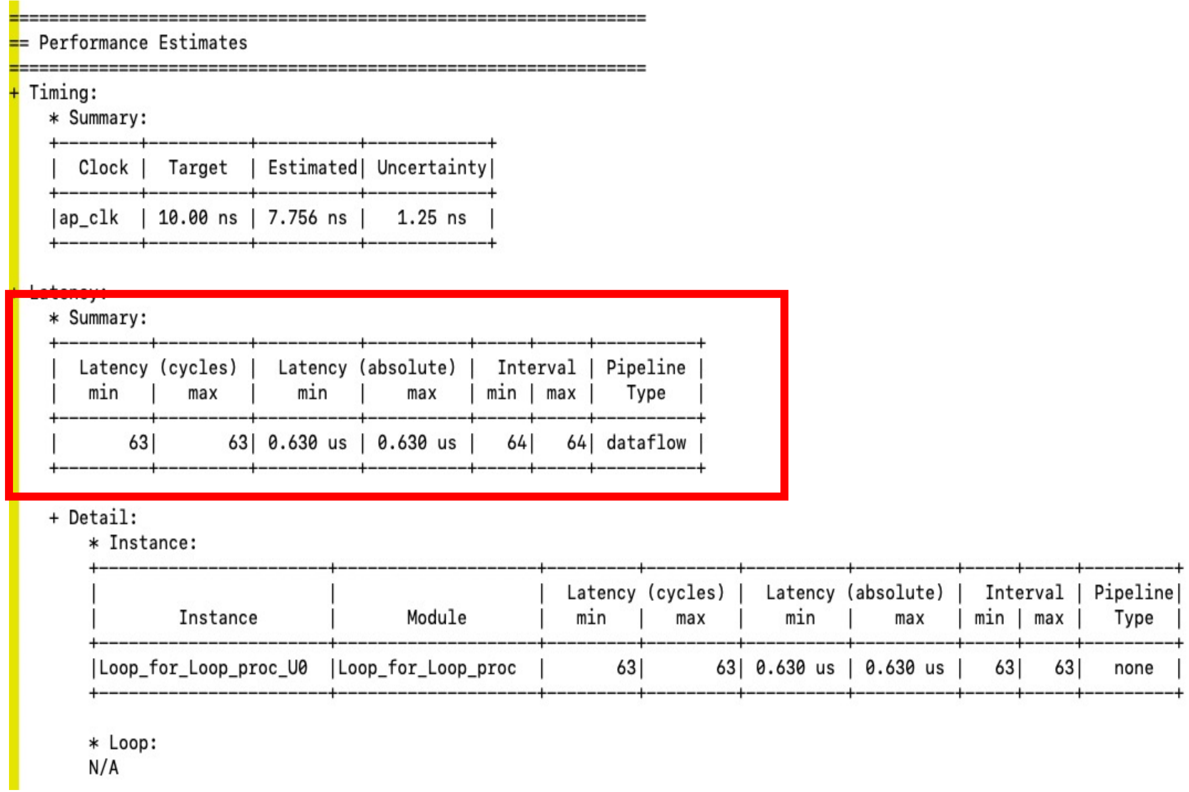
#pragma HLS pipeline II=2

With HLS PIPELINE II=2



```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |      122|      122| 1.220 us | 1.220 us |  122|  122|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +------------+---------+---------+-----------+-----------+-----------+------+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency  |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+-----------+-----------+-----------+------+----------+
        |- for_Loop  |      120|      120|          3|          2|          2|    60|    yes   |
        +------------+---------+---------+-----------+-----------+-----------+------+----------+
```

With DATAFLOW pipelining

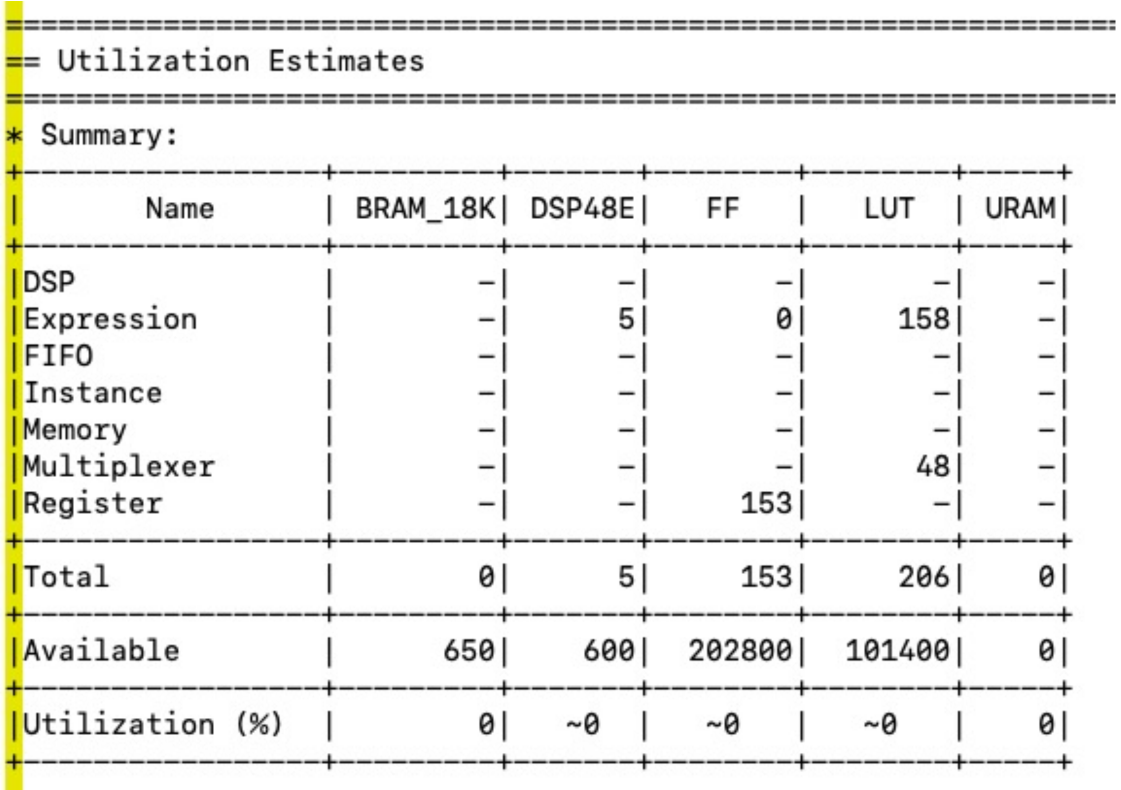


```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+----------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline |
    |   min   |   max   |    min   |    max   | min | max |   Type   |
    +---------+---------+----------+----------+-----+-----+----------+
    |       63|       63| 0.630 us | 0.630 us |   64|   64| dataflow |
    +---------+---------+----------+----------+-----+-----+----------+

    + Detail:
        * Instance:
        +----------------------+--------------------+---------+---------+----------+----------+-----+-----+---------+
        |                      |                    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
        |       Instance       |       Module       |   min   |   max   |    min   |    max   | min | max |   Type  |
        +----------------------+--------------------+---------+---------+----------+----------+-----+-----+---------+
        |Loop_for_Loop_proc_U0 |Loop_for_Loop_proc  |       63|       63| 0.630 us | 0.630 us |   63|   63|   none  |
        +----------------------+--------------------+---------+---------+----------+----------+-----+-----+---------+

        * Loop:
        N/A
```

Pragma HLS Pipeline

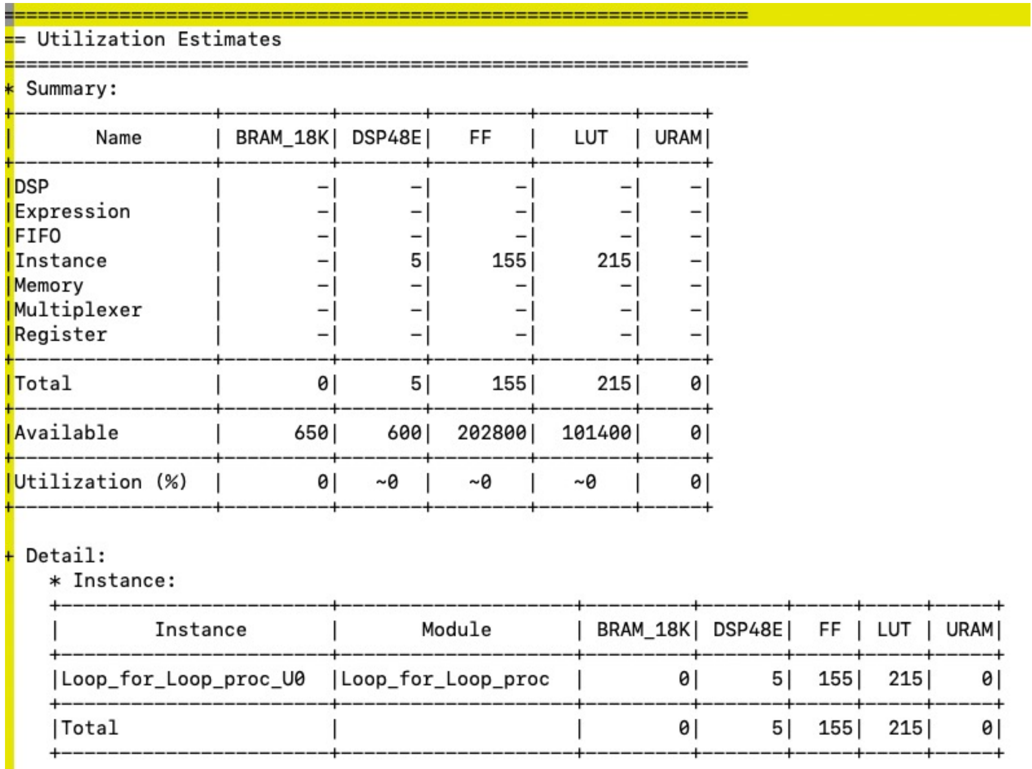
#pragma HLS pipeline II=2

With HLS PIPELINE II=2



```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     158|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      48|    -|
|Register         |        -|      -|     153|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     153|     206|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|   ~0  |   ~0   |   ~0   |    0|
+-----------------+---------+-------+--------+--------+-----+
```

With DATAFLOW pipelining



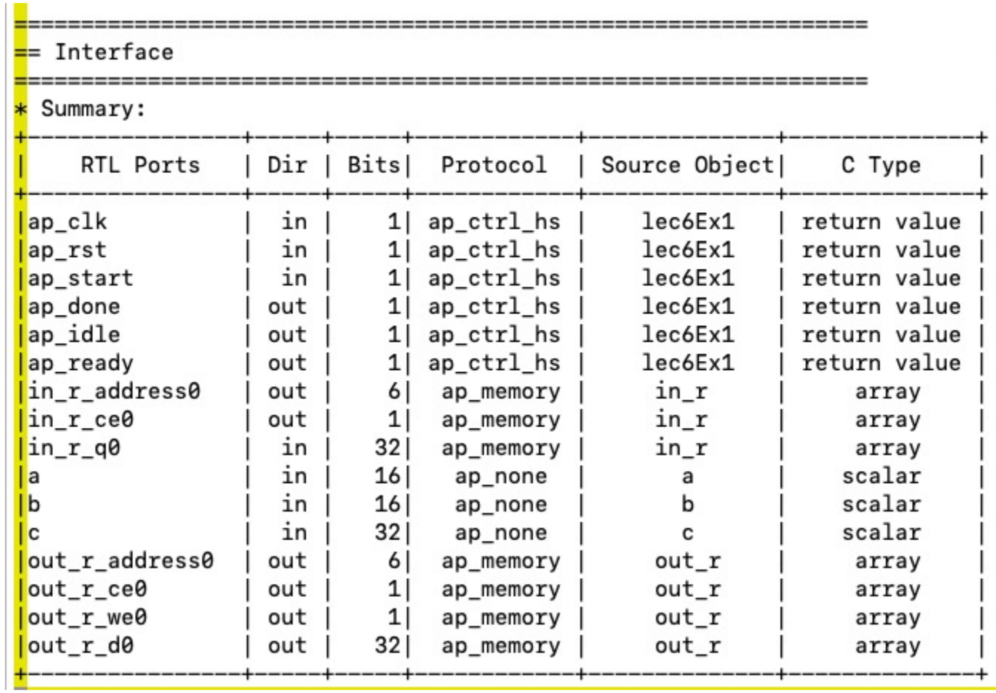
```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      -|       -|       -|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      5|     155|     215|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|       -|    -|
|Register         |        -|      -|       -|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     155|     215|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|   ~0  |   ~0   |   ~0   |    0|
+-----------------+---------+-------+--------+--------+-----+

+ Detail:
    * Instance:
    +----------------------+--------------------+---------+-------+-----+-----+-----+
    |       Instance       |       Module       | BRAM_18K| DSP48E|  FF | LUT | URAM|
    +----------------------+--------------------+---------+-------+-----+-----+-----+
    |Loop_for_Loop_proc_U0 |Loop_for_Loop_proc  |        0|      5|  155|  215|    0|
    +----------------------+--------------------+---------+-------+-----+-----+-----+
    |Total                 |                    |        0|      5|  155|  215|    0|
    +----------------------+--------------------+---------+-------+-----+-----+-----+
```

Pragma HLS Pipeline

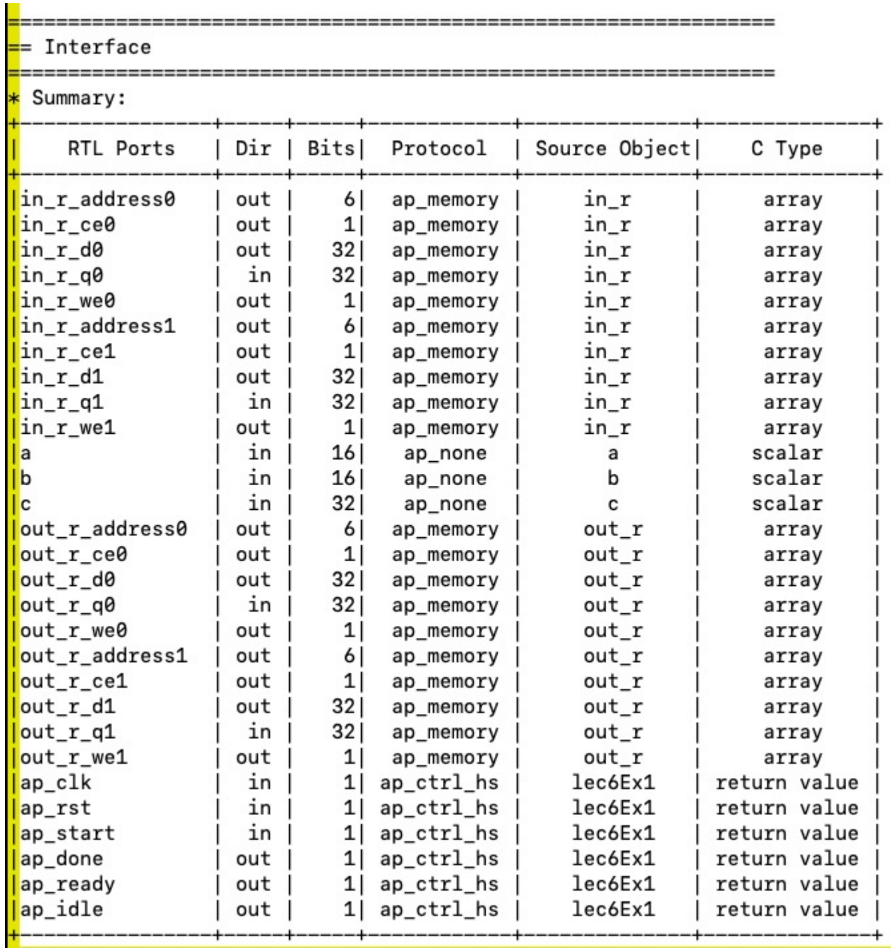
#pragma HLS pipeline II=2

With HLS PIPELINE II=2

With DATAFLOW pipelining



```
================================================================
== Interface
================================================================
* Summary:
+------------------+-----+-----+------------+--------------+--------------+
|     RTL Ports    | Dir | Bits|  Protocol  | Source Object|    C Type    |
+------------------+-----+-----+------------+--------------+--------------+
|ap_clk            |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_rst            |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_start          |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_done           | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_idle           | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_ready          | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|in_r_address0     | out |    6|  ap_memory |     in_r     |     array    |
|in_r_ce0          | out |    1|  ap_memory |     in_r     |     array    |
|in_r_q0           |  in |   32|  ap_memory |     in_r     |     array    |
|a                 |  in |   16|   ap_none  |       a      |    scalar    |
|b                 |  in |   16|   ap_none  |       b      |    scalar    |
|c                 |  in |   32|   ap_none  |       c      |    scalar    |
|out_r_address0    | out |    6|  ap_memory |     out_r    |     array    |
|out_r_ce0         | out |    1|  ap_memory |     out_r    |     array    |
|out_r_we0         | out |    1|  ap_memory |     out_r    |     array    |
|out_r_d0          | out |   32|  ap_memory |     out_r    |     array    |
+------------------+-----+-----+------------+--------------+--------------+
```



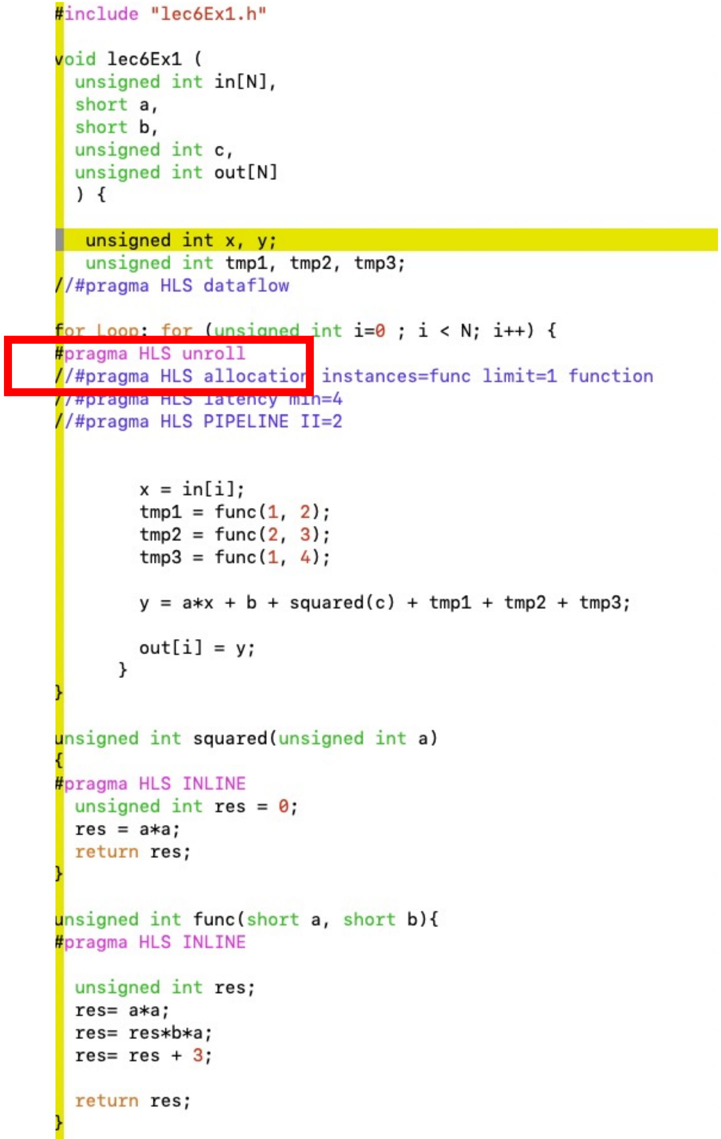
```
================================================================
== Interface
================================================================
* Summary:
+------------------+-----+-----+------------+--------------+--------------+
|     RTL Ports    | Dir | Bits|  Protocol  | Source Object|    C Type    |
+------------------+-----+-----+------------+--------------+--------------+
|in_r_address0     | out |    6|  ap_memory |     in_r     |     array    |
|in_r_ce0          | out |    1|  ap_memory |     in_r     |     array    |
|in_r_d0           | out |   32|  ap_memory |     in_r     |     array    |
|in_r_q0           |  in |   32|  ap_memory |     in_r     |     array    |
|in_r_we0          | out |    1|  ap_memory |     in_r     |     array    |
|in_r_address1     | out |    6|  ap_memory |     in_r     |     array    |
|in_r_ce1          | out |    1|  ap_memory |     in_r     |     array    |
|in_r_d1           | out |   32|  ap_memory |     in_r     |     array    |
|in_r_q1           |  in |   32|  ap_memory |     in_r     |     array    |
|in_r_we1          | out |    1|  ap_memory |     in_r     |     array    |
|a                 |  in |   16|   ap_none  |       a      |    scalar    |
|b                 |  in |   16|   ap_none  |       b      |    scalar    |
|c                 |  in |   32|   ap_none  |       c      |    scalar    |
|out_r_address0    | out |    6|  ap_memory |     out_r    |     array    |
|out_r_ce0         | out |    1|  ap_memory |     out_r    |     array    |
|out_r_d0          | out |   32|  ap_memory |     out_r    |     array    |
|out_r_q0          |  in |   32|  ap_memory |     out_r    |     array    |
|out_r_we0         | out |    1|  ap_memory |     out_r    |     array    |
|out_r_address1    | out |    6|  ap_memory |     out_r    |     array    |
|out_r_ce1         | out |    1|  ap_memory |     out_r    |     array    |
|out_r_d1          | out |   32|  ap_memory |     out_r    |     array    |
|out_r_q1          |  in |   32|  ap_memory |     out_r    |     array    |
|out_r_we1         | out |    1|  ap_memory |     out_r    |     array    |
|ap_clk            |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_rst            |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_start          |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_done           | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_ready          | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_idle           | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
+------------------+-----+-----+------------+--------------+--------------+
```

Pragma HLS unroll

#pragma HLS unroll

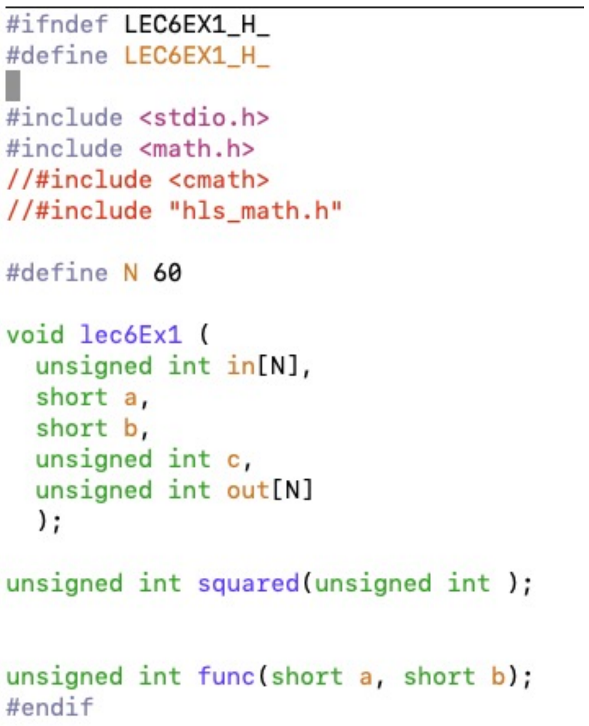
- Unroll loops to create multiple independent operations rather than a single collection of operations
- **UNROLL** pragma transforms loops by creating multiples copies of the loop body in the RTL design, which allows some or all loop iterations to occur in parallel
- Loops in the C/C++ functions are kept rolled by default
  - When loops are rolled, synthesis creates the logic for one iteration of the loop, and the RTL design executes this logic for each iteration of the loop in sequence
- **UNROLL** pragma allows the loop to be fully or partially unrolled
  - Fully unrolling the loop creates a copy of the loop body in the RTL for each loop iteration, so the entire loop can be run concurrently
  - Partially unrolling a loop lets you specify a factor *N*

Pragma HLS unroll

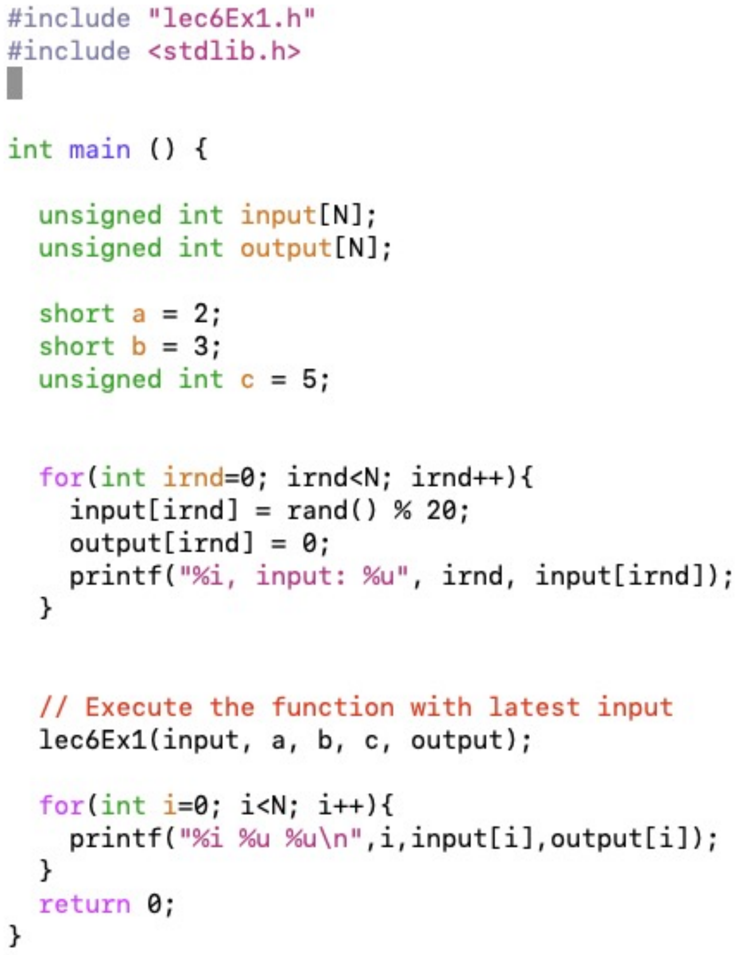


```
#include "lec6Ex1.h"

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  ) {

   unsigned int x, y;
   unsigned int tmp1, tmp2, tmp3;
//#pragma HLS dataflow

for_Loop: for (unsigned int i=0 ; i < N; i++) {
#pragma HLS unroll
//#pragma HLS allocation instances=func limit=1 function
//#pragma HLS latency min=4
//#pragma HLS PIPELINE II=2


        x = in[i];
        tmp1 = func(1, 2);
        tmp2 = func(2, 3);
        tmp3 = func(1, 4);

        y = a*x + b + squared(c) + tmp1 + tmp2 + tmp3;

        out[i] = y;
      }
}

unsigned int squared(unsigned int a)
{
#pragma HLS INLINE
  unsigned int res = 0;
  res = a*a;
  return res;
}

unsigned int func(short a, short b){
#pragma HLS INLINE

  unsigned int res;
  res= a*a;
  res= res*b*a;
  res= res + 3;

  return res;
}
```

#pragma HLS unroll



```
#ifndef LEC6EX1_H_
#define LEC6EX1_H_

#include <stdio.h>
#include <math.h>
//#include <cmath>
//#include "hls_math.h"

#define N 60

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  );

unsigned int squared(unsigned int );


unsigned int func(short a, short b);
#endif
```



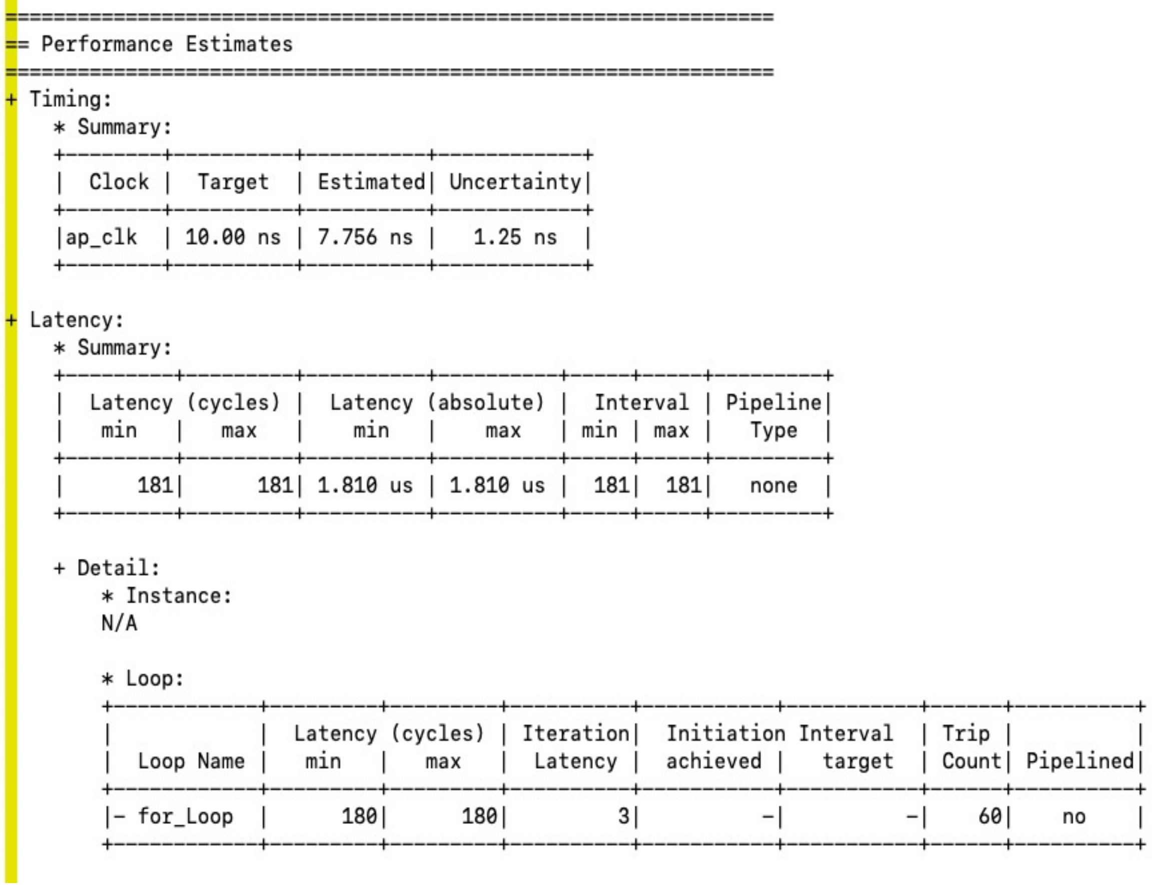
```
#include "lec6Ex1.h"
#include <stdlib.h>


int main () {

  unsigned int input[N];
  unsigned int output[N];

  short a = 2;
  short b = 3;
  unsigned int c = 5;


  for(int irnd=0; irnd<N; irnd++){
    input[irnd] = rand() % 20;
    output[irnd] = 0;
    printf("%i, input: %u", irnd, input[irnd]);
  }


  // Execute the function with latest input
  lec6Ex1(input, a, b, c, output);

  for(int i=0; i<N; i++){
    printf("%i %u %u\n",i,input[i],output[i]);
  }
  return 0;
}
```

Pragma HLS unroll

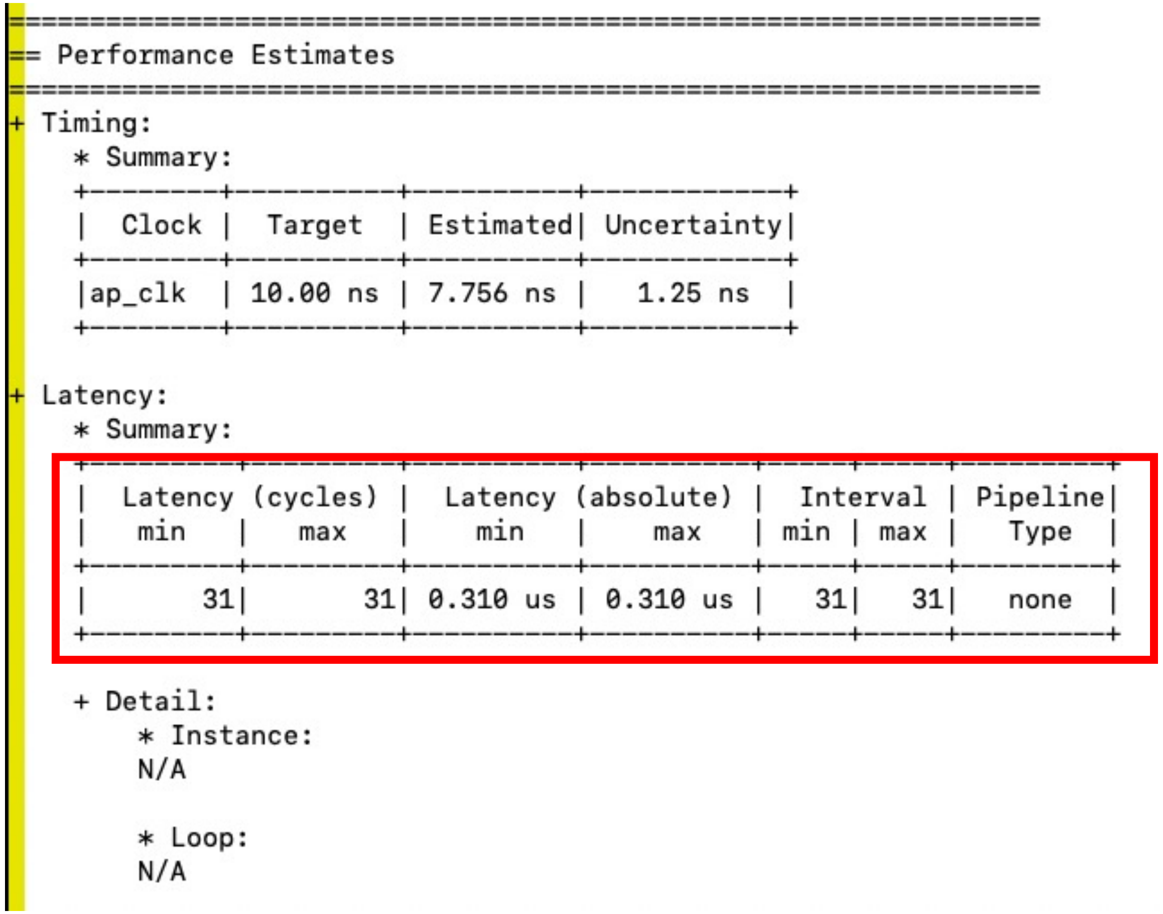
Without UNROLL For-loop

#pragma HLS unroll

With UNROLL For-loop



```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+---------+----------+------------+
    | Clock  |  Target | Estimated| Uncertainty|
    +--------+---------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+---------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |      181|      181| 1.810 us | 1.810 us |  181|  181|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |- for_Loop  |      180|      180|         3|          -|          -|    60|    no    |
        +------------+---------+---------+----------+-----------+-----------+------+----------+
```



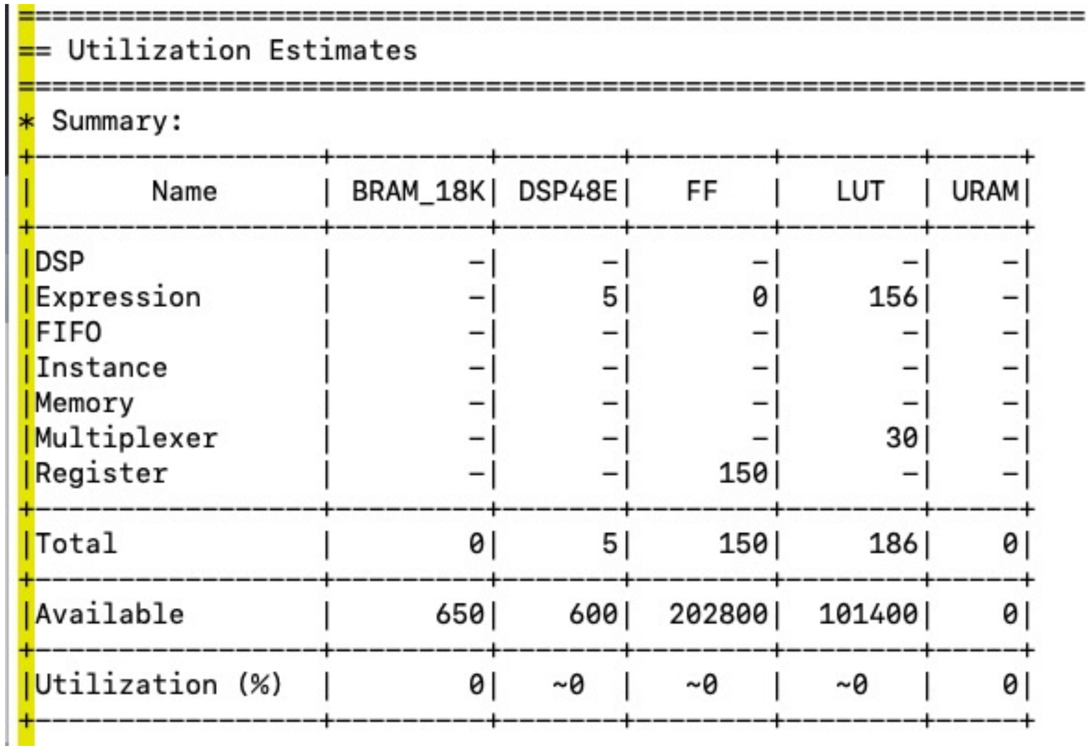
```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+---------+----------+------------+
    | Clock  |  Target | Estimated| Uncertainty|
    +--------+---------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+---------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |       31|       31| 0.310 us | 0.310 us |   31|   31|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        N/A
```

Pragma HLS unroll

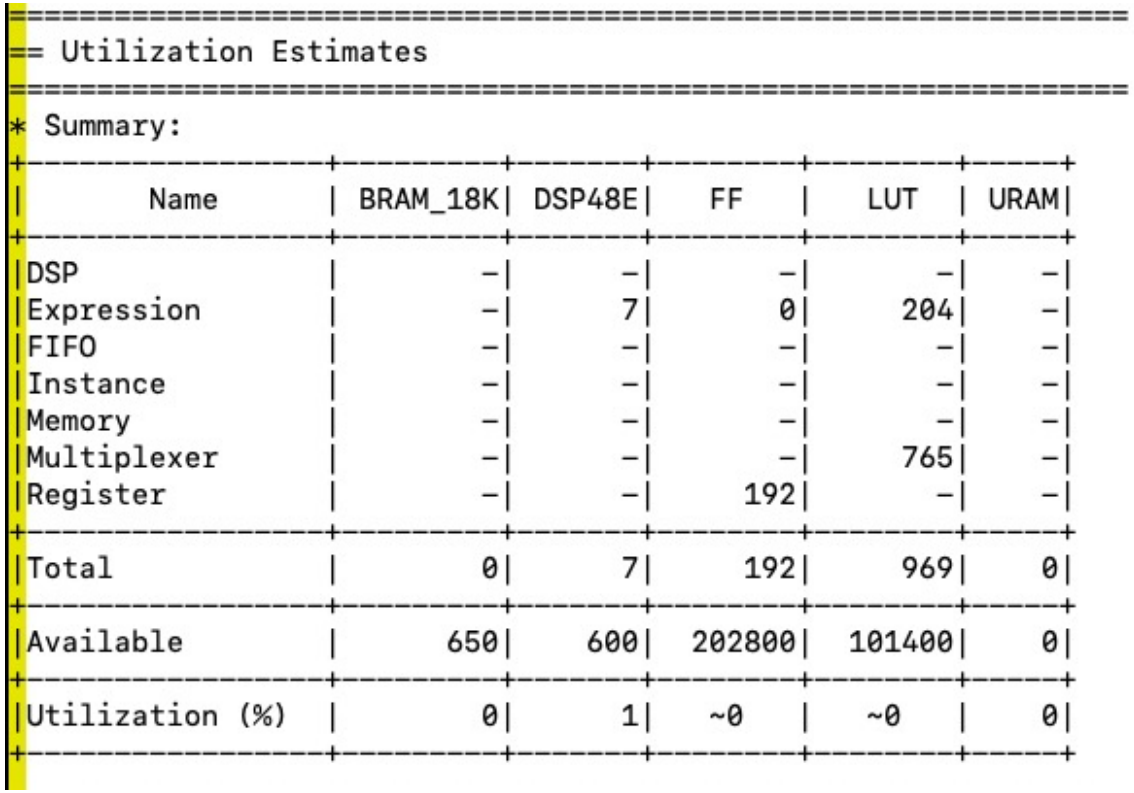
Without UNROLL For-loop

#pragma HLS unroll

With UNROLL For-loop



```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     156|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      30|    -|
|Register         |        -|      -|     150|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     150|     186|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|     ~0 |     ~0 |     ~0 |    0|
+-----------------+---------+-------+--------+--------+-----+
```



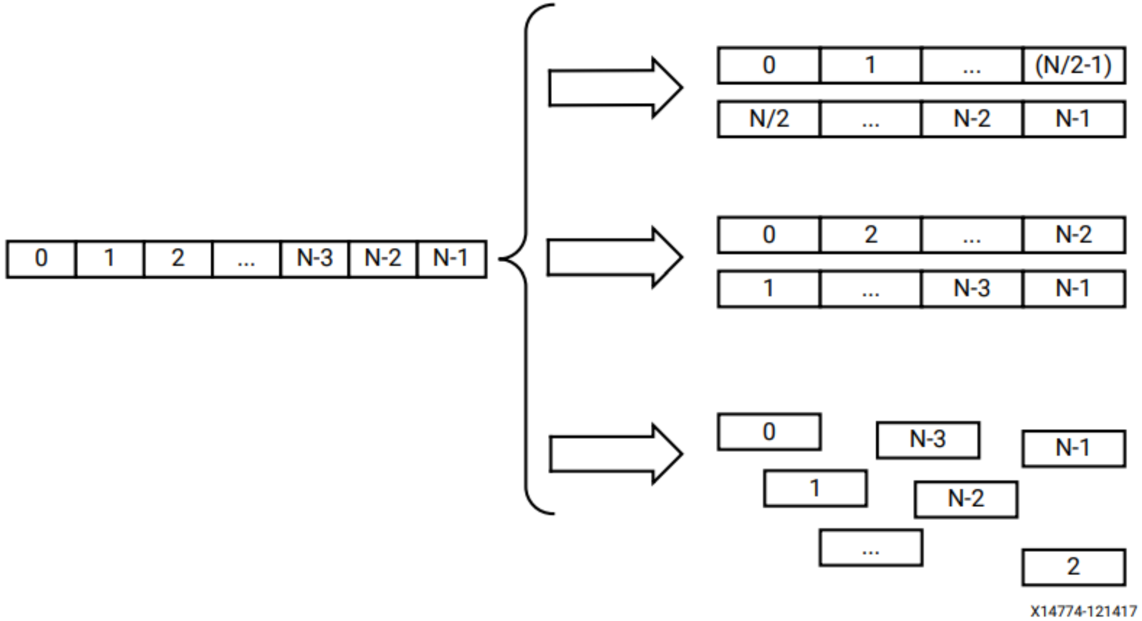
```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      7|       0|     204|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|     765|    -|
|Register         |        -|      -|     192|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      7|     192|     969|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|      1|     ~0 |     ~0 |    0|
+-----------------+---------+-------+--------+--------+-----+
```

Pragma HLS array_partition

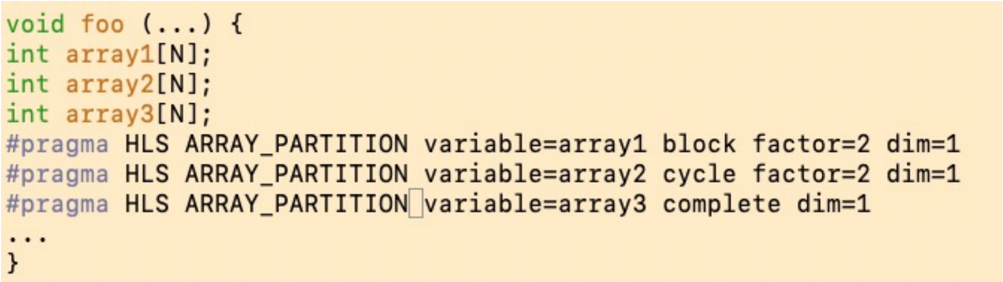
Array optimization

#pragma HLS array_partition variable=<name> <type> factor=<int> dim=<int>

- **Cyclic:** Cyclic partitioning creates smaller arrays by interleaving elements from the original array
- **Block:** Block partitioning creates smaller arrays from consecutive N-blocks of the original array
- **Complete:** Complete partitioning decomposes the array into individual elements
  - For a 1-D array, this corresponds to resolving a memory into individual registers (default <type>)

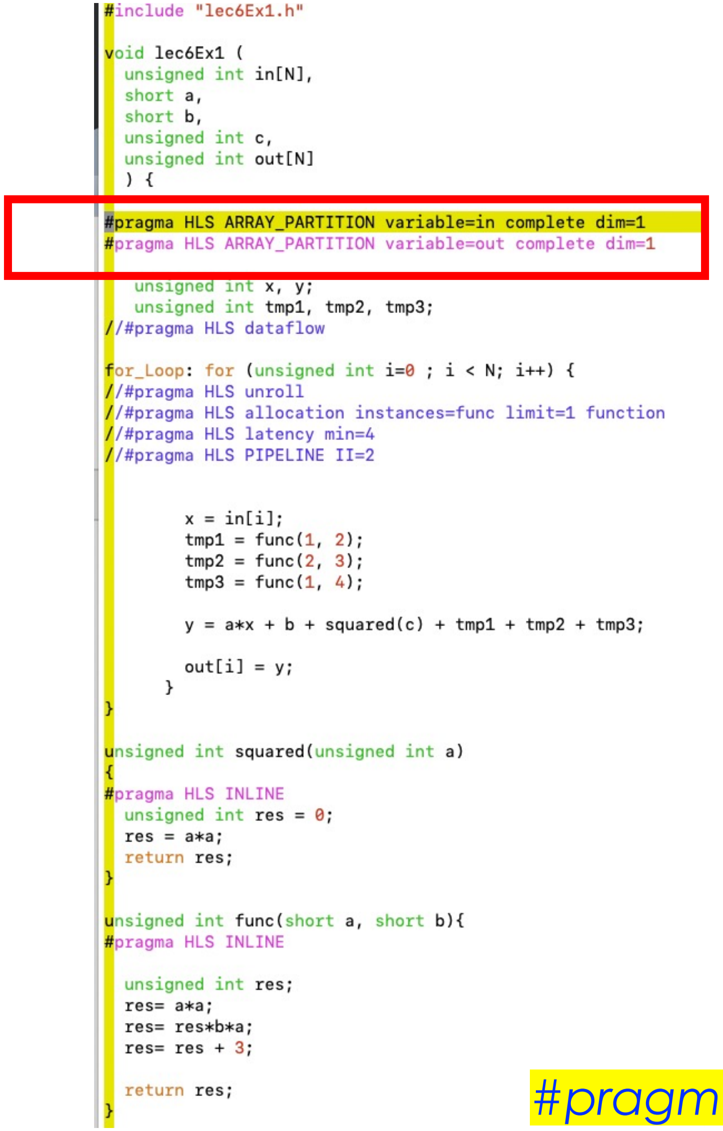


Figure

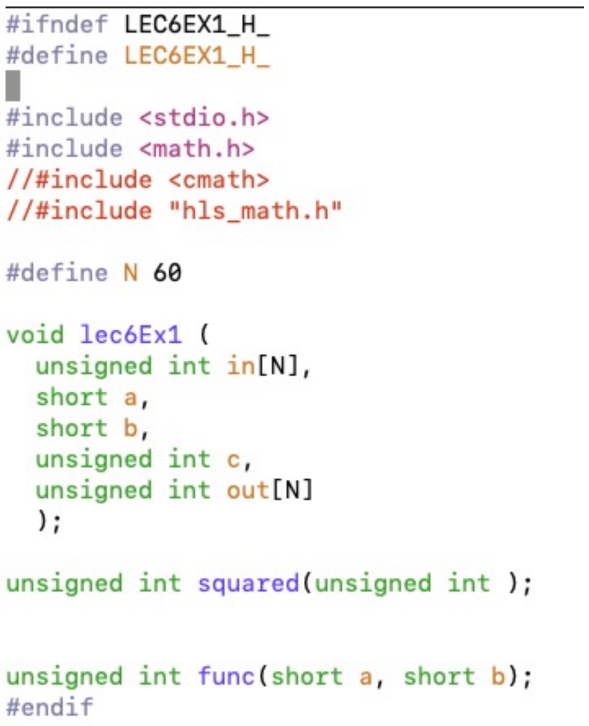


```
void foo (...) {
int array1[N];
int array2[N];
int array3[N];
#pragma HLS ARRAY_PARTITION variable=array1 block factor=2 dim=1
#pragma HLS ARRAY_PARTITION variable=array2 cycle factor=2 dim=1
#pragma HLS ARRAY_PARTITION variable=array3 complete dim=1
...
}
```

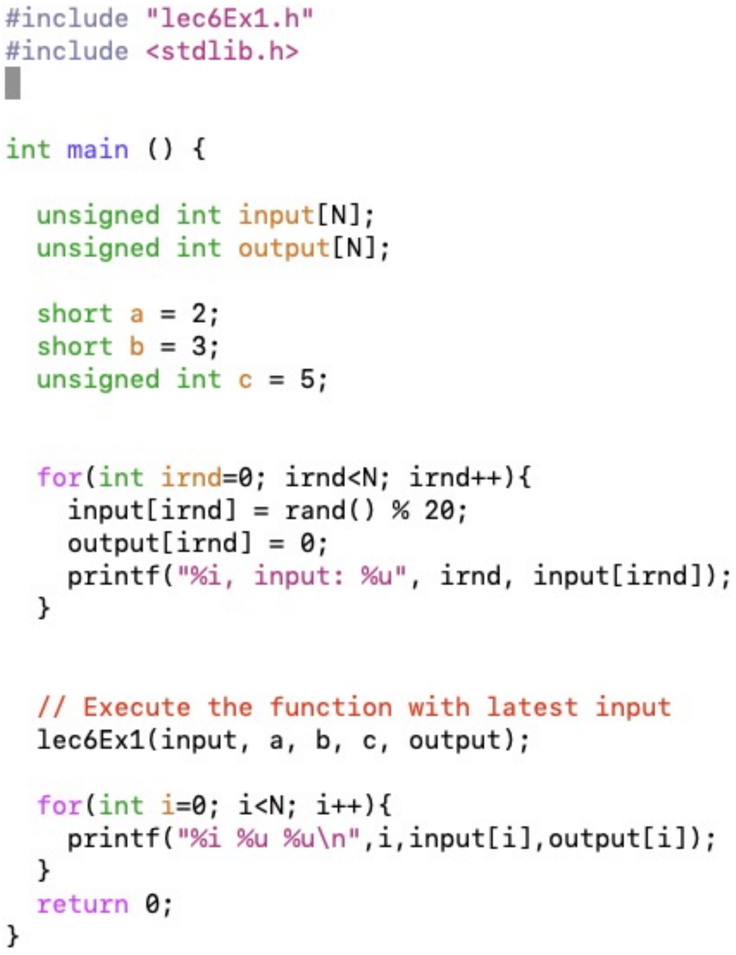
Pragma HLS array_partition



```
#include "lec6Ex1.h"

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  ) {

#pragma HLS ARRAY_PARTITION variable=in complete dim=1
#pragma HLS ARRAY_PARTITION variable=out complete dim=1

   unsigned int x, y;
   unsigned int tmp1, tmp2, tmp3;
//#pragma HLS dataflow

for_Loop: for (unsigned int i=0 ; i < N; i++) {
//#pragma HLS unroll
//#pragma HLS allocation instances=func limit=1 function
//#pragma HLS latency min=4
//#pragma HLS PIPELINE II=2


        x = in[i];
        tmp1 = func(1, 2);
        tmp2 = func(2, 3);
        tmp3 = func(1, 4);

        y = a*x + b + squared(c) + tmp1 + tmp2 + tmp3;

        out[i] = y;
      }
}

unsigned int squared(unsigned int a)
{
#pragma HLS INLINE
  unsigned int res = 0;
  res = a*a;
  return res;
}

unsigned int func(short a, short b){
#pragma HLS INLINE

  unsigned int res;
  res= a*a;
  res= res*b*a;
  res= res + 3;

  return res;
}
```



```
#ifndef LEC6EX1_H_
#define LEC6EX1_H_

#include <stdio.h>
#include <math.h>
//#include <cmath>
//#include "hls_math.h"

#define N 60

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  );

unsigned int squared(unsigned int );


unsigned int func(short a, short b);
#endif
```



```
#include "lec6Ex1.h"
#include <stdlib.h>


int main () {

  unsigned int input[N];
  unsigned int output[N];

  short a = 2;
  short b = 3;
  unsigned int c = 5;


  for(int irnd=0; irnd<N; irnd++){
    input[irnd] = rand() % 20;
    output[irnd] = 0;
    printf("%i, input: %u", irnd, input[irnd]);
  }


  // Execute the function with latest input
  lec6Ex1(input, a, b, c, output);

  for(int i=0; i<N; i++){
    printf("%i %u %u\n",i,input[i],output[i]);
  }
  return 0;
}
```

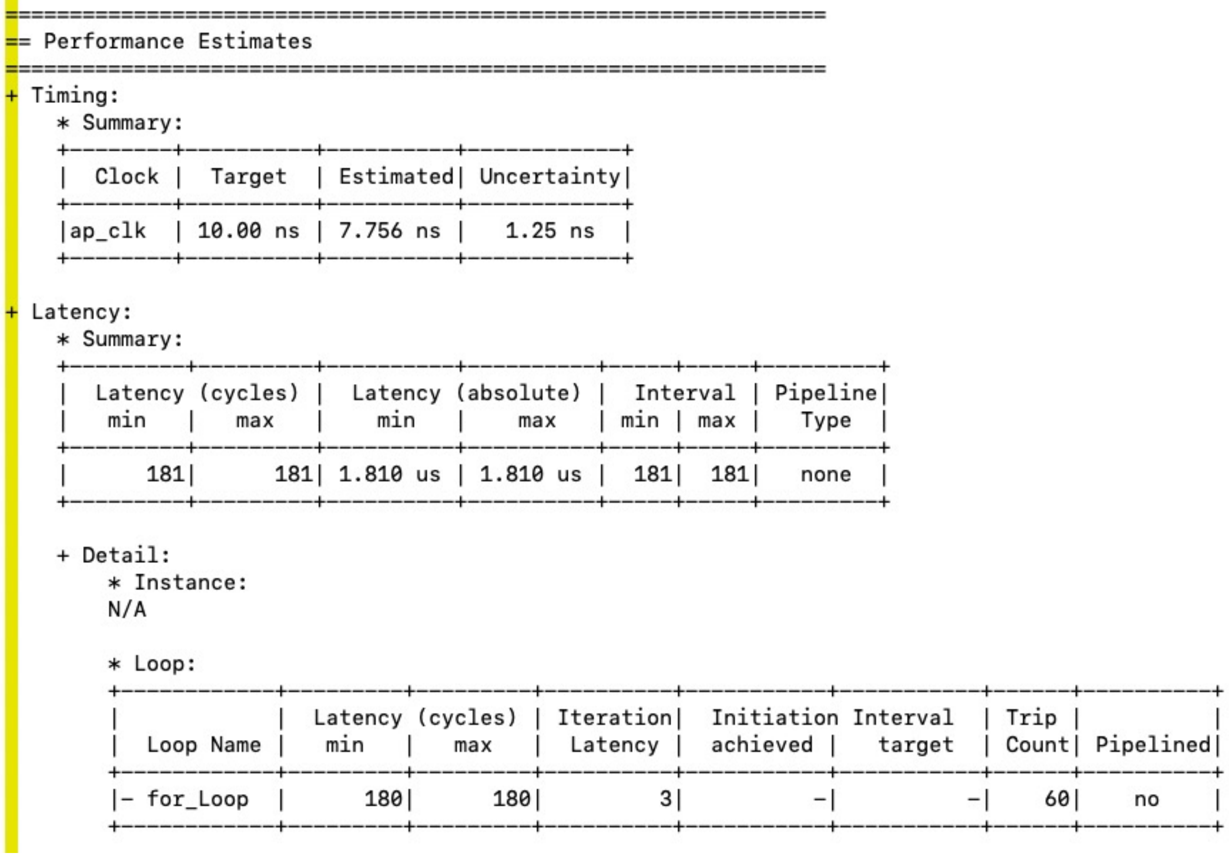
#pragma HLS array_partition variable=<name> <type> factor=<int> dim=<int>

Pragma HLS array_partition

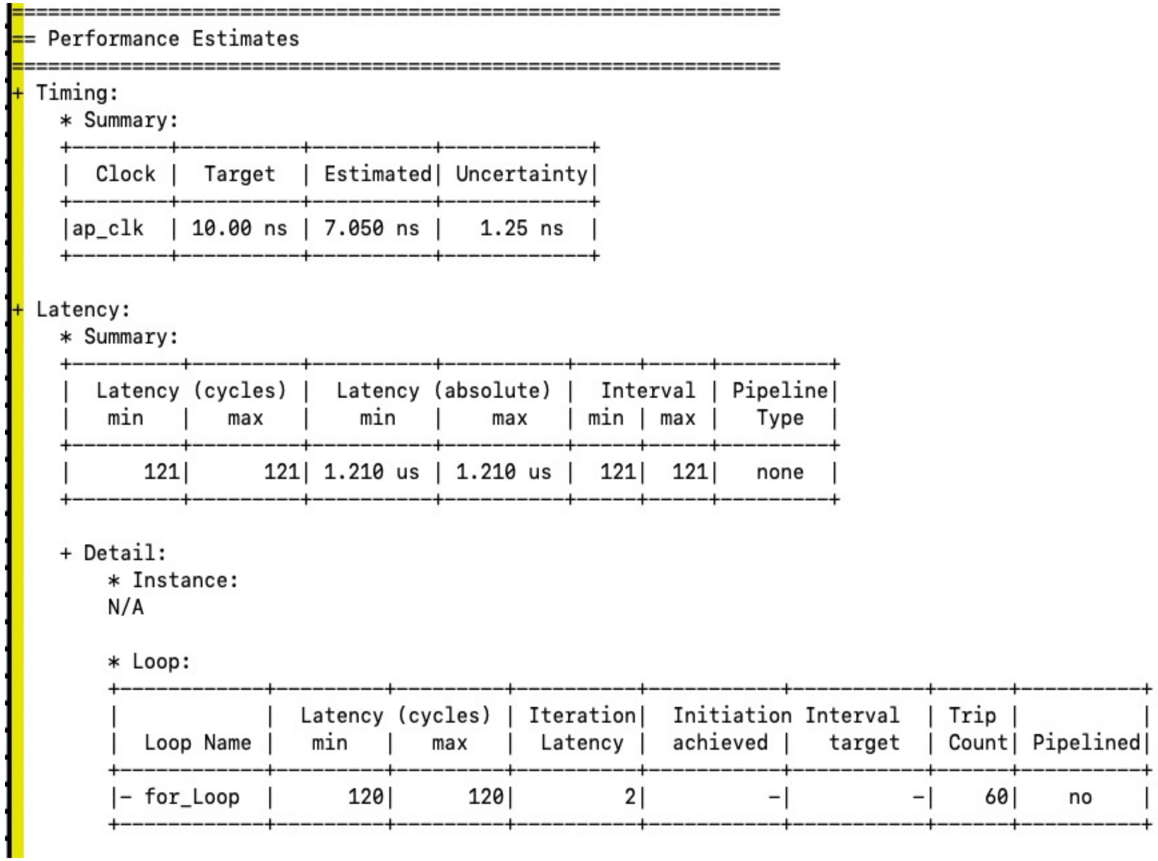
#pragma HLS array_partition variable=<name> <type> factor=<int> dim=<int>

Without Array partitioning

With Array Partitioning



```
=====================================================================
== Performance Estimates
=====================================================================
+ Timing:
    * Summary:
    +---------+----------+----------+------------+
    |  Clock  |  Target  | Estimated| Uncertainty|
    +---------+----------+----------+------------+
    |ap_clk   | 10.00 ns | 7.756 ns |   1.25 ns  |
    +---------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |      181|      181| 1.810 us | 1.810 us |  181|  181|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |- for_Loop  |      180|      180|         3|          -|          -|    60|    no    |
        +------------+---------+---------+----------+-----------+-----------+------+----------+
```

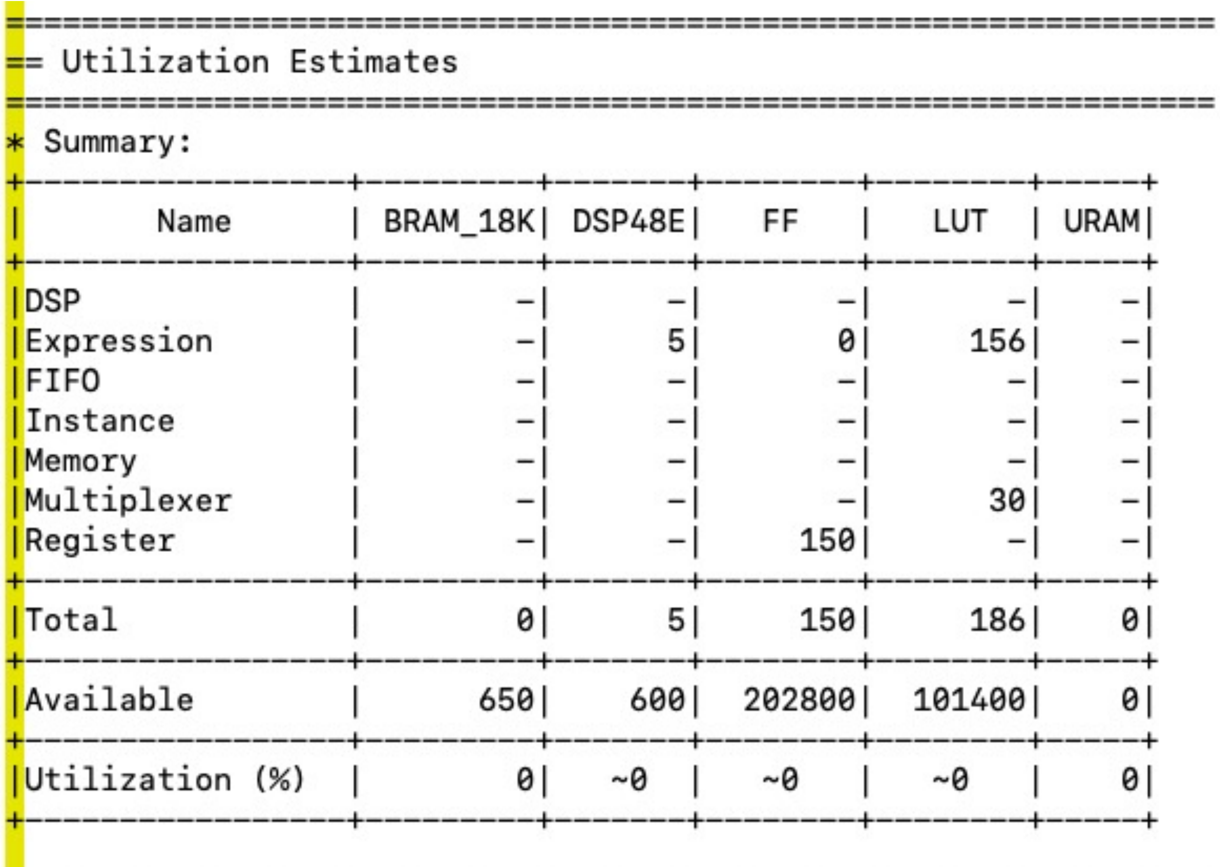


```
=====================================================================
== Performance Estimates
=====================================================================
+ Timing:
    * Summary:
    +---------+----------+----------+------------+
    |  Clock  |  Target  | Estimated| Uncertainty|
    +---------+----------+----------+------------+
    |ap_clk   | 10.00 ns | 7.050 ns |   1.25 ns  |
    +---------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |      121|      121| 1.210 us | 1.210 us |  121|  121|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+----------+-----------+-----------+------+----------+
        |- for_Loop  |      120|      120|         2|          -|          -|    60|    no    |
        +------------+---------+---------+----------+-----------+-----------+------+----------+
```

Pragma HLS array_partition

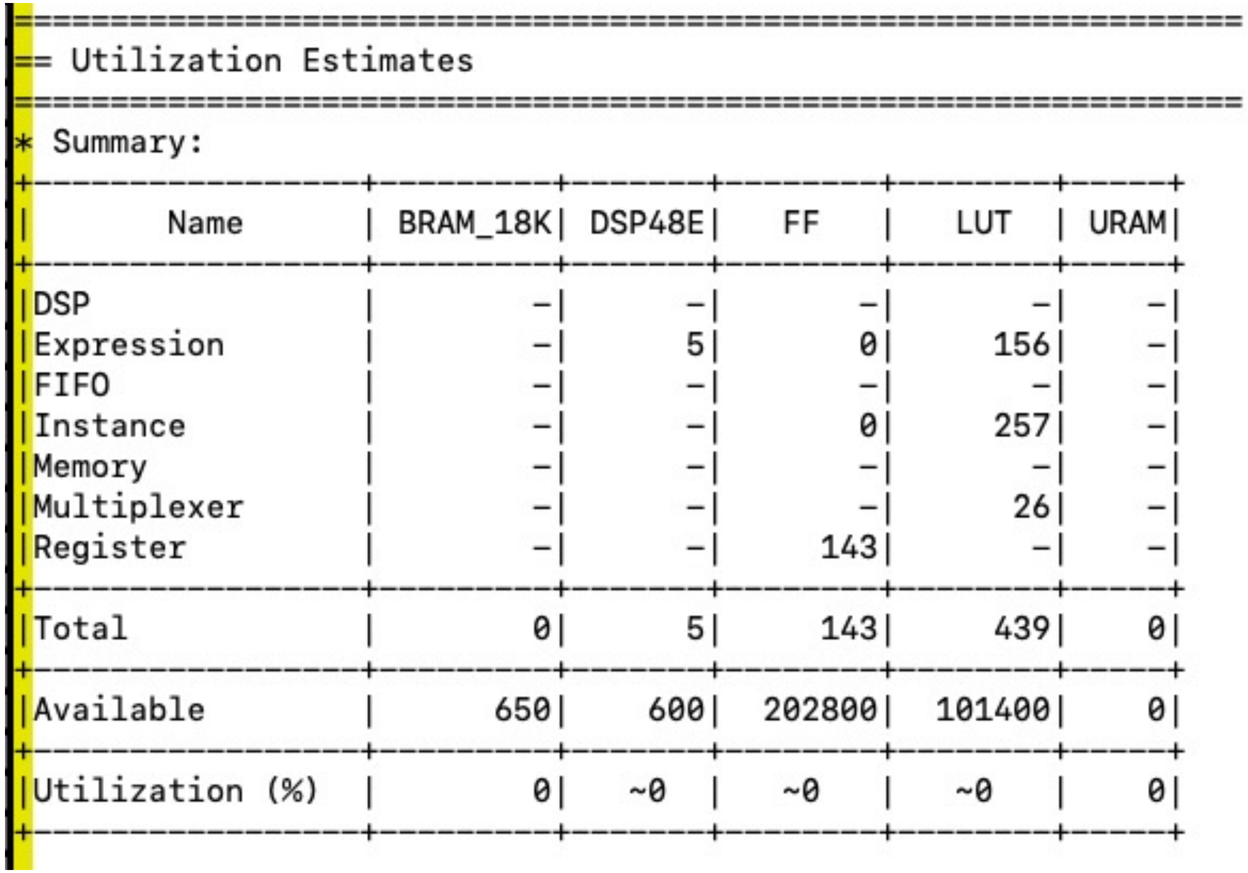
#pragma HLS array_partition variable=<name> <type> factor=<int> dim=<int>

Without Array partitioning



```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     156|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      30|    -|
|Register         |        -|      -|     150|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     150|     186|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|   ~0  |   ~0   |   ~0   |    0|
+-----------------+---------+-------+--------+--------+-----+
```

With Array Partitioning

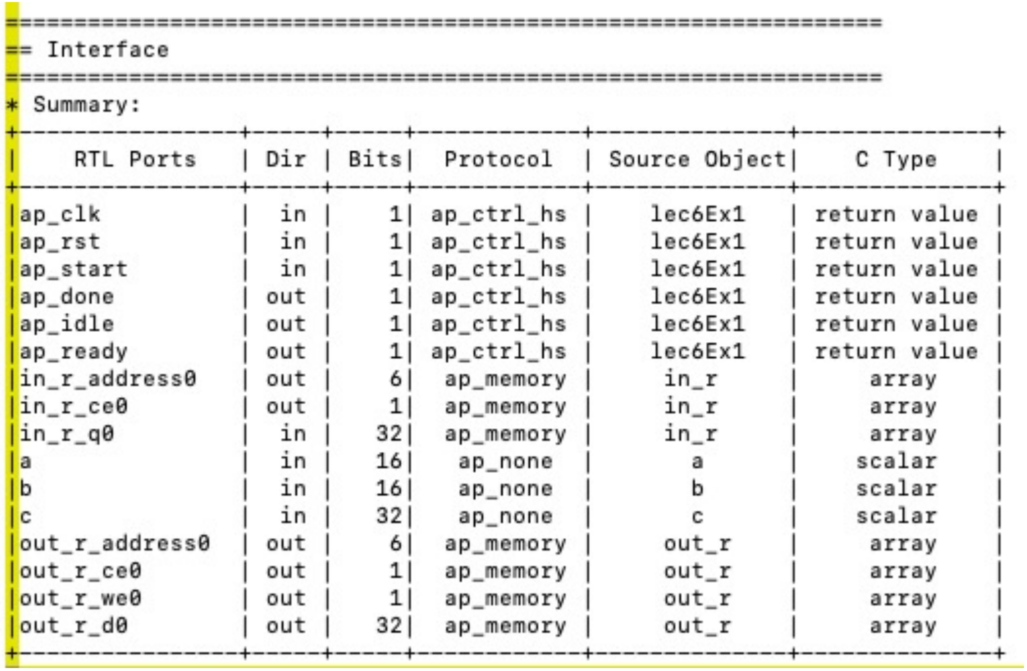


```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     156|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       0|     257|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      26|    -|
|Register         |        -|      -|     143|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     143|     439|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|   ~0  |   ~0   |   ~0   |    0|
+-----------------+---------+-------+--------+--------+-----+
```

Pragma HLS array_partition

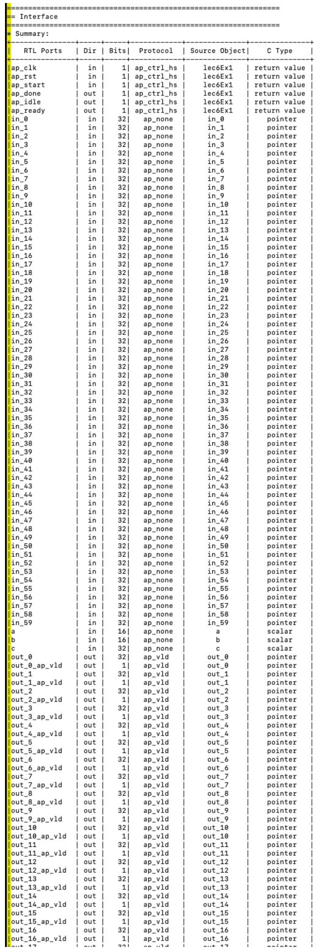
#pragma HLS array_partition variable=<name> <type> factor=<int> dim=<int>

Without Array partitioning



```
================================================================
== Interface
================================================================
* Summary:
+-----------------+-----+-----+------------+--------------+--------------+
|    RTL Ports    | Dir | Bits|  Protocol  | Source Object|    C Type    |
+-----------------+-----+-----+------------+--------------+--------------+
|ap_clk           |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_rst           |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_start         |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_done          | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_idle          | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_ready         | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|in_r_address0    | out |    6|  ap_memory |     in_r     |     array    |
|in_r_ce0         | out |    1|  ap_memory |     in_r     |     array    |
|in_r_q0          |  in |   32|  ap_memory |     in_r     |     array    |
|a                |  in |   16|   ap_none  |       a      |    scalar    |
|b                |  in |   16|   ap_none  |       b      |    scalar    |
|c                |  in |   32|   ap_none  |       c      |    scalar    |
|out_r_address0   | out |    6|  ap_memory |     out_r    |     array    |
|out_r_ce0        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_we0        | out |    1|  ap_memory |     out_r    |     array    |
|out_r_d0         | out |   32|  ap_memory |     out_r    |     array    |
+-----------------+-----+-----+------------+--------------+--------------+
```

With Array Partitioning



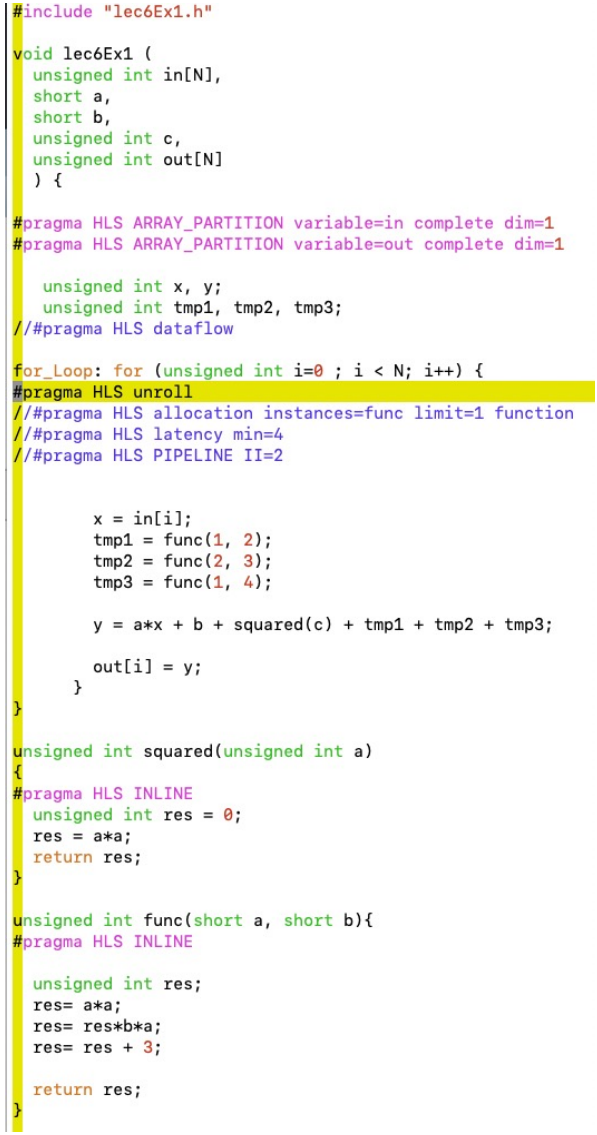
```
================================================================
== Interface
================================================================
* Summary:
+-----------------+-----+-----+------------+--------------+--------------+
|    RTL Ports    | Dir | Bits|  Protocol  | Source Object|    C Type    |
+-----------------+-----+-----+------------+--------------+--------------+
|ap_clk           |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_rst           |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_start         |  in |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_done          | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_idle          | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|ap_ready         | out |    1| ap_ctrl_hs |    lec6Ex1   | return value |
|in_0             |  in |   32|   ap_none  |     in_0     |    pointer   |
|in_1             |  in |   32|   ap_none  |     in_1     |    pointer   |
|in_2             |  in |   32|   ap_none  |     in_2     |    pointer   |
|in_3             |  in |   32|   ap_none  |     in_3     |    pointer   |
|in_4             |  in |   32|   ap_none  |     in_4     |    pointer   |
|in_5             |  in |   32|   ap_none  |     in_5     |    pointer   |
|in_6             |  in |   32|   ap_none  |     in_6     |    pointer   |
|in_7             |  in |   32|   ap_none  |     in_7     |    pointer   |
|in_8             |  in |   32|   ap_none  |     in_8     |    pointer   |
|in_9             |  in |   32|   ap_none  |     in_9     |    pointer   |
|in_10            |  in |   32|   ap_none  |     in_10    |    pointer   |
|in_11            |  in |   32|   ap_none  |     in_11    |    pointer   |
|in_12            |  in |   32|   ap_none  |     in_12    |    pointer   |
|in_13            |  in |   32|   ap_none  |     in_13    |    pointer   |
|in_14            |  in |   32|   ap_none  |     in_14    |    pointer   |
|in_15            |  in |   32|   ap_none  |     in_15    |    pointer   |
|in_16            |  in |   32|   ap_none  |     in_16    |    pointer   |
|in_17            |  in |   32|   ap_none  |     in_17    |    pointer   |
|in_18            |  in |   32|   ap_none  |     in_18    |    pointer   |
|in_19            |  in |   32|   ap_none  |     in_19    |    pointer   |
|in_20            |  in |   32|   ap_none  |     in_20    |    pointer   |
|in_21            |  in |   32|   ap_none  |     in_21    |    pointer   |
|in_22            |  in |   32|   ap_none  |     in_22    |    pointer   |
|in_23            |  in |   32|   ap_none  |     in_23    |    pointer   |
|in_24            |  in |   32|   ap_none  |     in_24    |    pointer   |
|in_25            |  in |   32|   ap_none  |     in_25    |    pointer   |
|in_26            |  in |   32|   ap_none  |     in_26    |    pointer   |
|in_27            |  in |   32|   ap_none  |     in_27    |    pointer   |
|in_28            |  in |   32|   ap_none  |     in_28    |    pointer   |
|in_29            |  in |   32|   ap_none  |     in_29    |    pointer   |
|in_30            |  in |   32|   ap_none  |     in_30    |    pointer   |
|in_31            |  in |   32|   ap_none  |     in_31    |    pointer   |
|in_32            |  in |   32|   ap_none  |     in_32    |    pointer   |
|in_33            |  in |   32|   ap_none  |     in_33    |    pointer   |
|in_34            |  in |   32|   ap_none  |     in_34    |    pointer   |
|in_35            |  in |   32|   ap_none  |     in_35    |    pointer   |
|in_36            |  in |   32|   ap_none  |     in_36    |    pointer   |
|in_37            |  in |   32|   ap_none  |     in_37    |    pointer   |
|in_38            |  in |   32|   ap_none  |     in_38    |    pointer   |
|in_39            |  in |   32|   ap_none  |     in_39    |    pointer   |
|in_40            |  in |   32|   ap_none  |     in_40    |    pointer   |
|in_41            |  in |   32|   ap_none  |     in_41    |    pointer   |
|in_42            |  in |   32|   ap_none  |     in_42    |    pointer   |
|in_43            |  in |   32|   ap_none  |     in_43    |    pointer   |
|in_44            |  in |   32|   ap_none  |     in_44    |    pointer   |
|in_45            |  in |   32|   ap_none  |     in_45    |    pointer   |
|in_46            |  in |   32|   ap_none  |     in_46    |    pointer   |
|in_47            |  in |   32|   ap_none  |     in_47    |    pointer   |
|in_48            |  in |   32|   ap_none  |     in_48    |    pointer   |
|in_49            |  in |   32|   ap_none  |     in_49    |    pointer   |
|in_50            |  in |   32|   ap_none  |     in_50    |    pointer   |
|in_51            |  in |   32|   ap_none  |     in_51    |    pointer   |
|in_52            |  in |   32|   ap_none  |     in_52    |    pointer   |
|in_53            |  in |   32|   ap_none  |     in_53    |    pointer   |
|in_54            |  in |   32|   ap_none  |     in_54    |    pointer   |
|in_55            |  in |   32|   ap_none  |     in_55    |    pointer   |
|in_56            |  in |   32|   ap_none  |     in_56    |    pointer   |
|in_57            |  in |   32|   ap_none  |     in_57    |    pointer   |
|in_58            |  in |   32|   ap_none  |     in_58    |    pointer   |
|in_59            |  in |   32|   ap_none  |     in_59    |    pointer   |
|a                |  in |   16|   ap_none  |       a      |    scalar    |
|b                |  in |   16|   ap_none  |       b      |    scalar    |
|c                |  in |   32|   ap_none  |       c      |    scalar    |
|out_0            | out |   32|   ap_vld   |     out_0    |    pointer   |
|out_0_ap_vld     | out |    1|   ap_vld   |     out_0    |    pointer   |
|out_1            | out |   32|   ap_vld   |     out_1    |    pointer   |
|out_1_ap_vld     | out |    1|   ap_vld   |     out_1    |    pointer   |
|out_2            | out |   32|   ap_vld   |     out_2    |    pointer   |
|out_2_ap_vld     | out |    1|   ap_vld   |     out_2    |    pointer   |
|out_3            | out |   32|   ap_vld   |     out_3    |    pointer   |
|out_3_ap_vld     | out |    1|   ap_vld   |     out_3    |    pointer   |
|out_4            | out |   32|   ap_vld   |     out_4    |    pointer   |
|out_4_ap_vld     | out |    1|   ap_vld   |     out_4    |    pointer   |
|out_5            | out |   32|   ap_vld   |     out_5    |    pointer   |
|out_5_ap_vld     | out |    1|   ap_vld   |     out_5    |    pointer   |
|out_6            | out |   32|   ap_vld   |     out_6    |    pointer   |
|out_6_ap_vld     | out |    1|   ap_vld   |     out_6    |    pointer   |
|out_7            | out |   32|   ap_vld   |     out_7    |    pointer   |
|out_7_ap_vld     | out |    1|   ap_vld   |     out_7    |    pointer   |
|out_8            | out |   32|   ap_vld   |     out_8    |    pointer   |
|out_8_ap_vld     | out |    1|   ap_vld   |     out_8    |    pointer   |
|out_9            | out |   32|   ap_vld   |     out_9    |    pointer   |
|out_9_ap_vld     | out |    1|   ap_vld   |     out_9    |    pointer   |
|out_10           | out |   32|   ap_vld   |    out_10    |    pointer   |
|out_10_ap_vld    | out |    1|   ap_vld   |    out_10    |    pointer   |
|out_11           | out |   32|   ap_vld   |    out_11    |    pointer   |
|out_11_ap_vld    | out |    1|   ap_vld   |    out_11    |    pointer   |
|out_12           | out |   32|   ap_vld   |    out_12    |    pointer   |
|out_12_ap_vld    | out |    1|   ap_vld   |    out_12    |    pointer   |
|out_13           | out |   32|   ap_vld   |    out_13    |    pointer   |
|out_13_ap_vld    | out |    1|   ap_vld   |    out_13    |    pointer   |
|out_14           | out |   32|   ap_vld   |    out_14    |    pointer   |
|out_14_ap_vld    | out |    1|   ap_vld   |    out_14    |    pointer   |
|out_15           | out |   32|   ap_vld   |    out_15    |    pointer   |
|out_15_ap_vld    | out |    1|   ap_vld   |    out_15    |    pointer   |
|out_16           | out |   32|   ap_vld   |    out_16    |    pointer   |
|out_16_ap_vld    | out |    1|   ap_vld   |    out_16    |    pointer   |
|out_17           | out |   32|   ap_vld   |    out_17    |    pointer   |
```

TAC-HEP 2023

Combination of Pragmas

ARRAY Partitioning + UNROLLING For loop

For loop unrolling + Array Partitioning



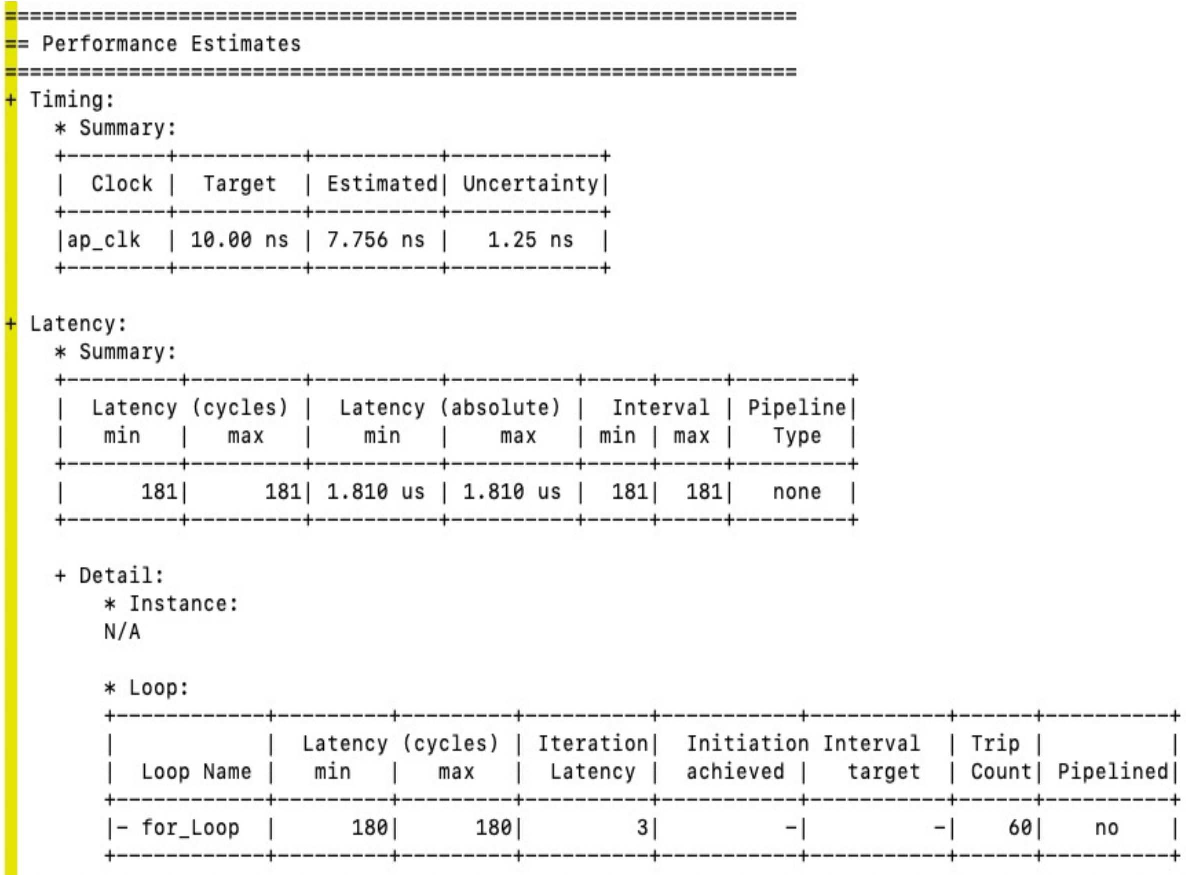
```
#include "lec6Ex1.h"

void lec6Ex1 (
  unsigned int in[N],
  short a,
  short b,
  unsigned int c,
  unsigned int out[N]
  ) {

#pragma HLS ARRAY_PARTITION variable=in complete dim=1
#pragma HLS ARRAY_PARTITION variable=out complete dim=1

   unsigned int x, y;
   unsigned int tmp1, tmp2, tmp3;
//#pragma HLS dataflow

for_Loop: for (unsigned int i=0 ; i < N; i++) {
#pragma HLS unroll
//#pragma HLS allocation instances=func limit=1 function
//#pragma HLS latency min=4
//#pragma HLS PIPELINE II=2


        x = in[i];
        tmp1 = func(1, 2);
        tmp2 = func(2, 3);
        tmp3 = func(1, 4);

        y = a*x + b + squared(c) + tmp1 + tmp2 + tmp3;

        out[i] = y;
      }
}

unsigned int squared(unsigned int a)
{
#pragma HLS INLINE
  unsigned int res = 0;
  res = a*a;
  return res;
}

unsigned int func(short a, short b){
#pragma HLS INLINE

  unsigned int res;
  res= a*a;
  res= res*b*a;
  res= res + 3;

  return res;
}
```

```
#pragma HLS ARRAY_PARTITION variable=in complete dim=1
#pragma HLS ARRAY_PARTITION variable=out complete dim=1
```

```
#pragma HLS unroll
```

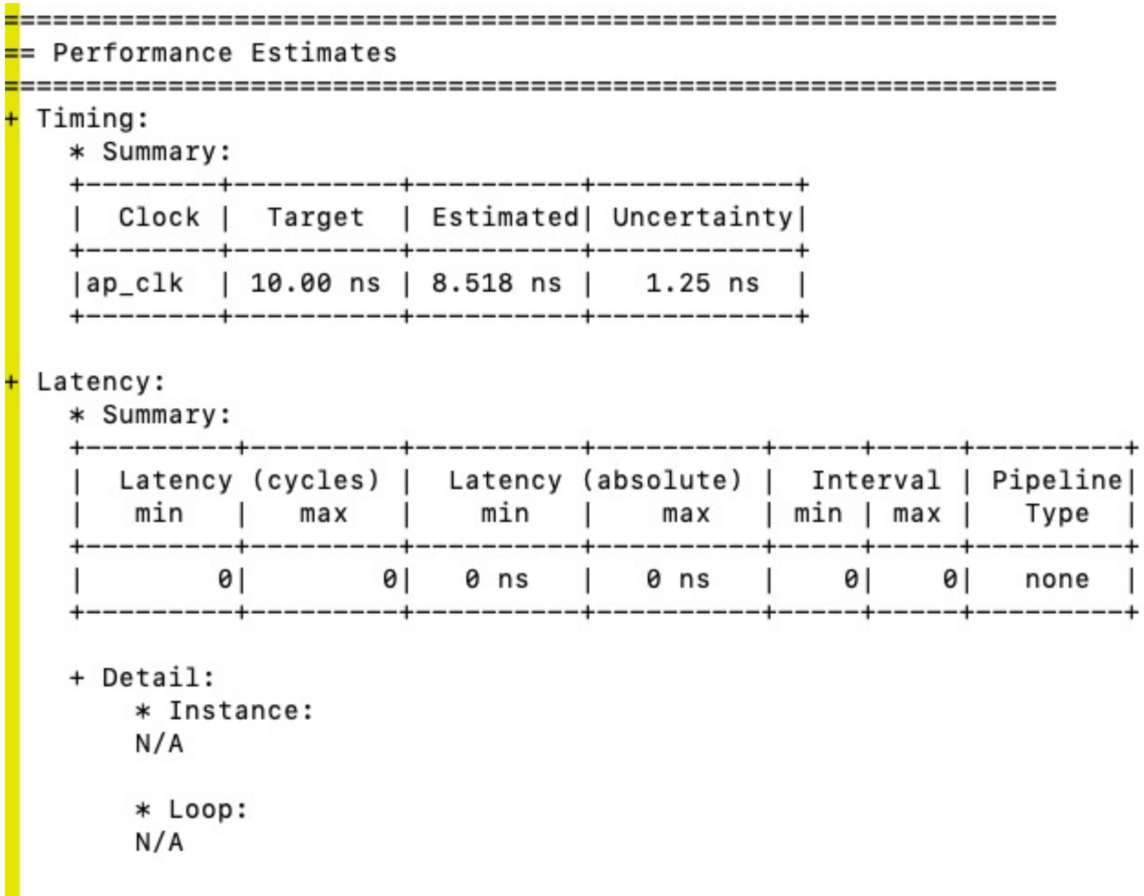
For loop unrolling + Array Partitioning

Without Pragma



```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 7.756 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |      181|      181| 1.810 us | 1.810 us |  181|  181|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        +------------+---------+---------+----------+-----------+-----------+-----+----------+
        |            |  Latency (cycles) | Iteration|  Initiation Interval  | Trip |          |
        |  Loop Name |   min   |   max   |  Latency |  achieved |   target  | Count| Pipelined|
        +------------+---------+---------+----------+-----------+-----------+-----+----------+
        |- for_Loop  |      180|      180|         3|          -|          -|   60|    no    |
        +------------+---------+---------+----------+-----------+-----------+-----+----------+
```

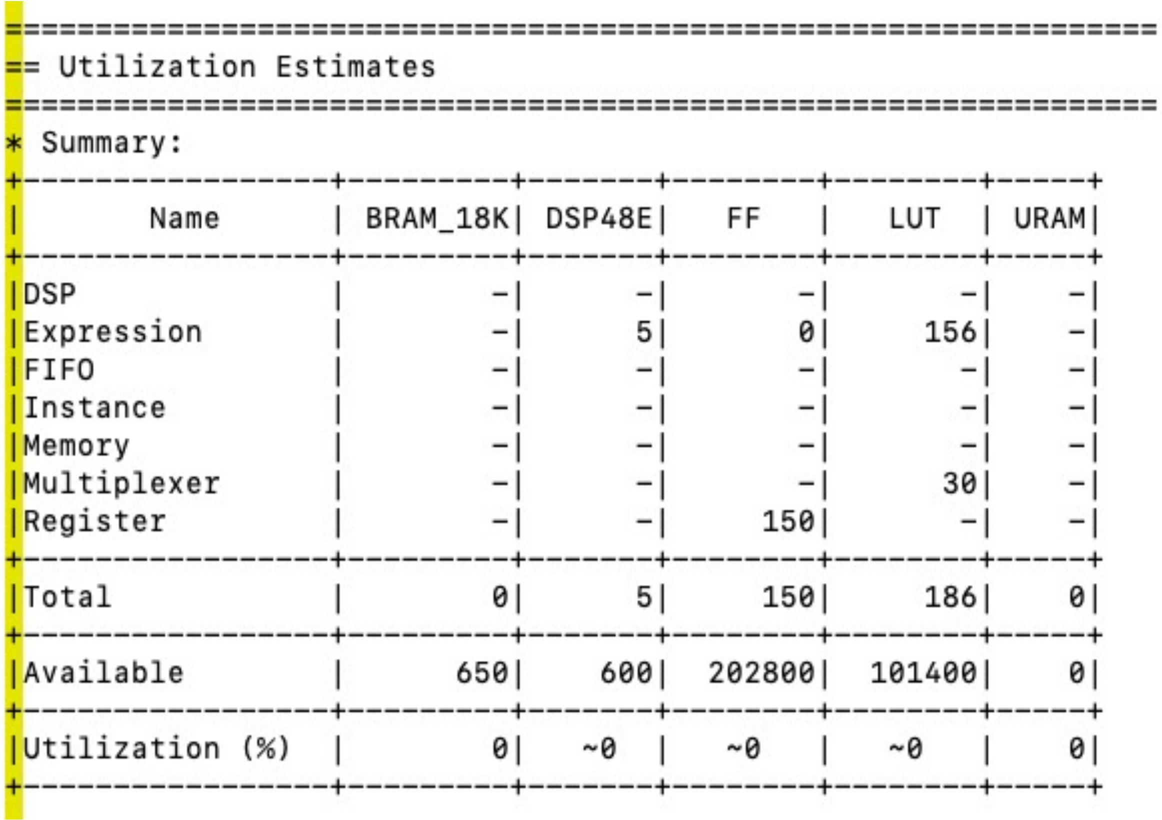
With Pragmas



```
================================================================
== Performance Estimates
================================================================
+ Timing:
    * Summary:
    +--------+----------+----------+------------+
    |  Clock |  Target  | Estimated| Uncertainty|
    +--------+----------+----------+------------+
    |ap_clk  | 10.00 ns | 8.518 ns |   1.25 ns  |
    +--------+----------+----------+------------+

+ Latency:
    * Summary:
    +---------+---------+----------+----------+-----+-----+---------+
    |  Latency (cycles) |  Latency (absolute) |  Interval | Pipeline|
    |   min   |   max   |    min   |    max   | min | max |   Type  |
    +---------+---------+----------+----------+-----+-----+---------+
    |        0|        0|   0 ns   |   0 ns   |    0|    0|   none  |
    +---------+---------+----------+----------+-----+-----+---------+

    + Detail:
        * Instance:
        N/A

        * Loop:
        N/A
```

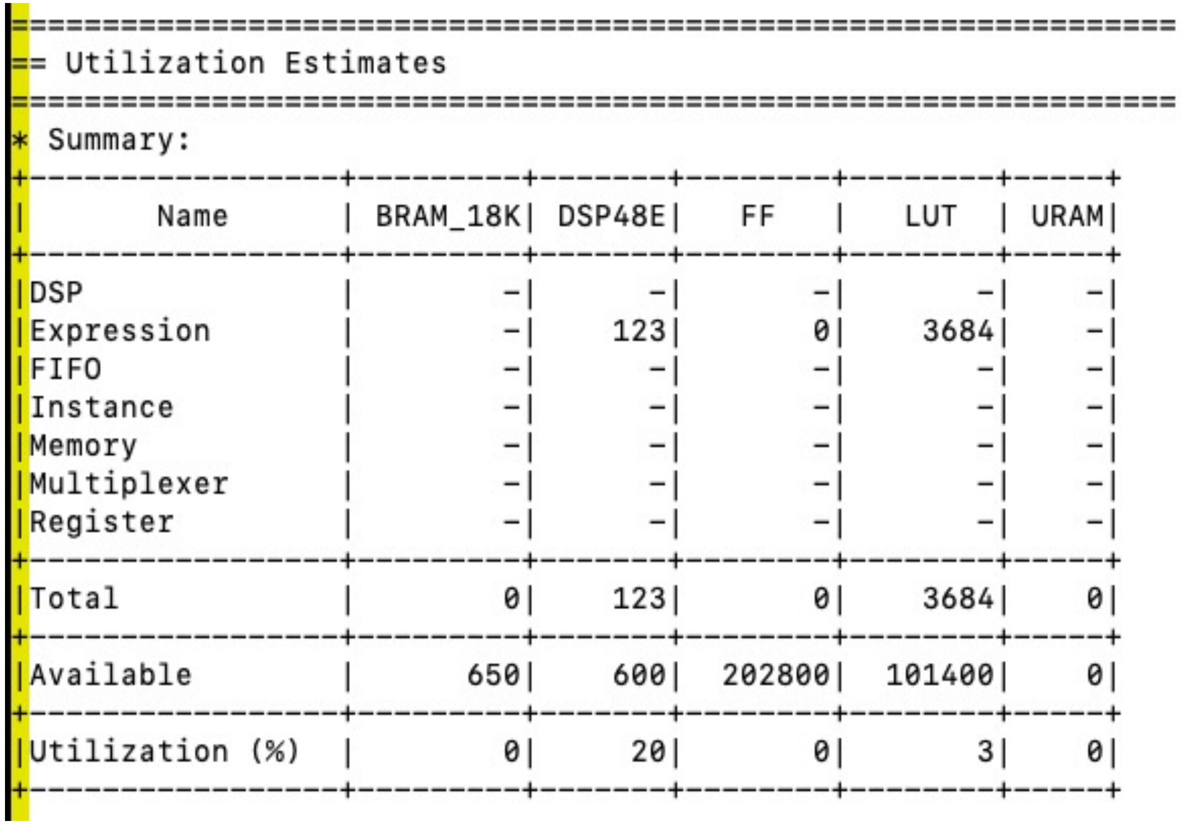
For loop unrolling + Array Partitioning

Without Pragma



```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|      5|       0|     156|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|      30|    -|
|Register         |        -|      -|     150|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|      5|     150|     186|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|    ~0 |     ~0 |     ~0 |    0|
+-----------------+---------+-------+--------+--------+-----+
```

With Pragmas



```
================================================================
== Utilization Estimates
================================================================
* Summary:
+-----------------+---------+-------+--------+--------+-----+
|       Name      | BRAM_18K| DSP48E|   FF   |   LUT  | URAM|
+-----------------+---------+-------+--------+--------+-----+
|DSP              |        -|      -|       -|       -|    -|
|Expression       |        -|    123|       0|    3684|    -|
|FIFO             |        -|      -|       -|       -|    -|
|Instance         |        -|      -|       -|       -|    -|
|Memory           |        -|      -|       -|       -|    -|
|Multiplexer      |        -|      -|       -|       -|    -|
|Register         |        -|      -|       -|       -|    -|
+-----------------+---------+-------+--------+--------+-----+
|Total            |        0|    123|       0|    3684|    0|
+-----------------+---------+-------+--------+--------+-----+
|Available        |      650|    600|  202800|  101400|    0|
+-----------------+---------+-------+--------+--------+-----+
|Utilization (%)  |        0|     20|       0|       3|    0|
+-----------------+---------+-------+--------+--------+-----+
```

Is it a good optimization?

Summary

- Pragmas are important to implement a design in best possible ways
- There are a lot of pragmas to help the design implementation
- Be careful with the choice of pragma’s to avoid conflicts
- Different pragma’s help improve different aspects of design or performance parameters

Assignment Week-4

1. Do a matrix multiplication of two 1-dimensional arrays – A[N]*B[N], where N > 5
   a) Report synthesis results without any pragma directives
   b) Add as many pragma directives possible
      i. Report any conflicts (if reported in logs) between two pragmas

2. Compare the analysis perspective (Performance) for different case shared today
3. For Array_partitioning, instead of using complete, use block and cyclic with different factors

Questions?

TAC-HEP 2023

Acknowledgement

Lectures are compiled using content from Xilinx’s public pages or different user guides

TAC-HEP 2023

Additional material

Assignment submission

- Where to submit:
  - https://pages.hep.wisc.edu/~varuns/assignments/TAC-HEP/
- Use your login machine credentials
- Submit one file per week
- Try to submit by following week’s Tuesday

Correct Time

From 03.28.2023 onwards

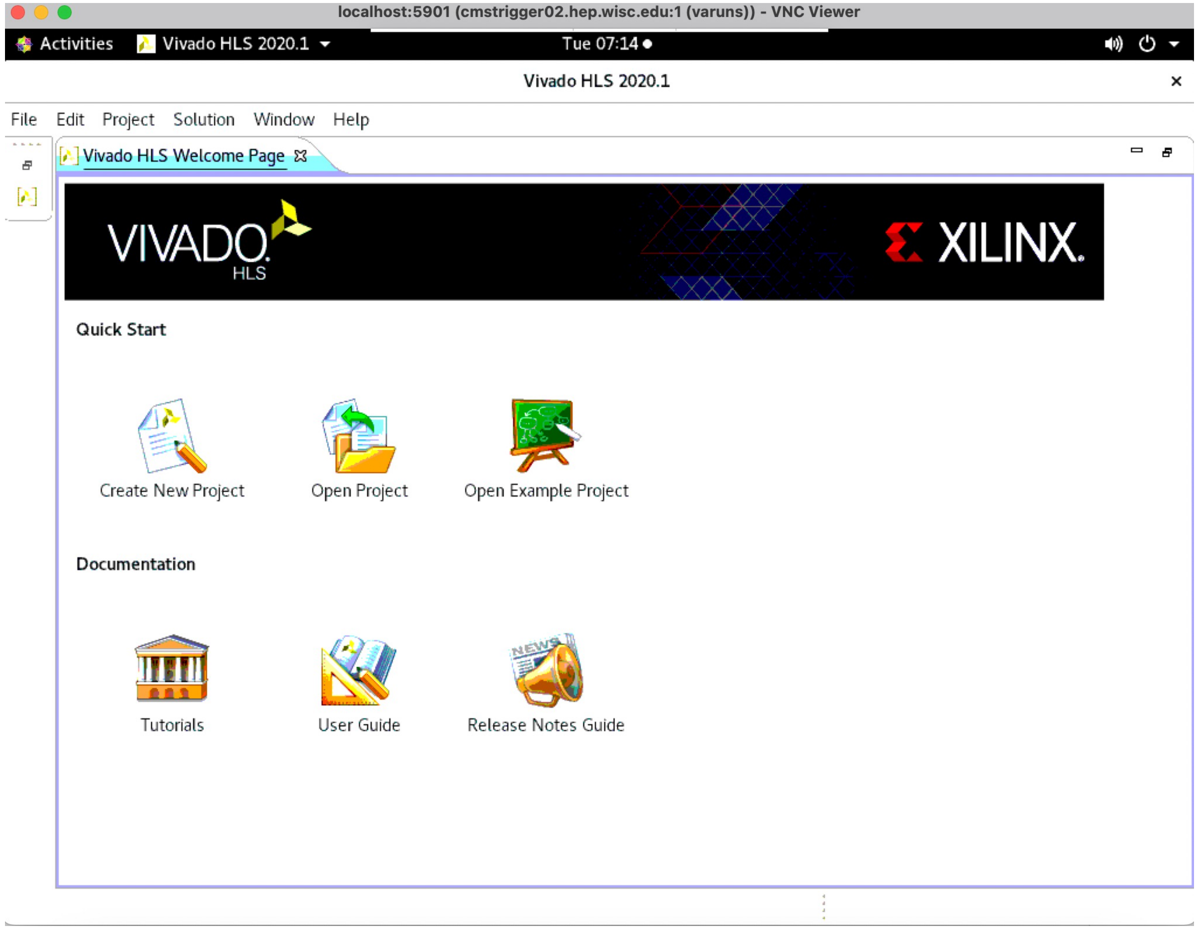
- Tuesdays: 9:00-10:00 CT / 10:00-11:00 ET / 16:00-17:00 CET
- Wednesday: 11:00-12:00 CT / 12:00-13:00 ET / 18:00-19:00 CET

Jargons

- **ICs - Integrated chip:** assembly of hundreds of millions of transistors on a minor chip
- **PCB:** Printed Circuit Board
- **LUT - Look Up Table aka 'logic'** - generic functions on small bitwidth inputs. Combine many to build the algorithm
- **FF - Flip Flops** - control the flow of data with the clock pulse. Used to build the pipeline and achieve high throughput
- **DSP - Digital Signal Processor** - performs multiplication and other arithmetic in the FPGA
- **BRAM - Block RAM** - hardened RAM resource. More efficient memories than using LUTs for more than a few elements
- **PCIe or PCI-E - Peripheral Component Interconnect Express**: is a serial expansion bus standard for connecting a computer to one or more peripheral devices
- **InfiniBand** is a computer networking communications standard used in high-performance computing that features very high throughput and very low latency
- **HLS -** High Level Synthesis - compiler for C, C++, SystemC into FPGA IP cores
- **DRCs -** Design Rule Checks
- **HDL** - Hardware Description Language - low level language for describing circuits
- **RTL** - Register Transfer Level - the very low level description of the function and connection of logic gates
- **FIFO** – First In First Out memory
- **Latency** - time between starting processing and receiving the result
  - Measured in clock cycles or seconds
- **II - Initiation Interval** - time from accepting first input to accepting next input

Reminder: Steps to follow

- **Step-1: Creating a New Project/Opening an existing project**
- **Step-2: Validating the C-source code**
- **Step-3: High Level Synthesis**
- **Step-4: RTL Verification**
- **Step-5: IP Creation**



Assignment Week-3

- Use target device: **xc7k160tfbg484-2**
- Clock period of 10ns

1. Execute the code (lec5Ex2.tcl) using CLI (slide-25) and compare the results with GUI results for C-Simulation, C-Synthesis

2. Vary following parameters for two cases: high and very high values and compare with 1 for both CLI and GUI
   - Variable: “samples”
   - Variable: “N”

3. Run example lec3Ex2a