ASSIGNMENT-1

ARCHITECTURES OF HARDWARE ACCELERATION

Parallel Insertion Sorting

N Adharsh - PES1UG21EC161

Akshay Anand - PES1UG21EC029

Nandan vadeyar - PES1UG21EC089

Akash ravi bhat - PES1UG21EC025

Abhishek A shetty – PES1UG21EC008

HLS Cpp code:

```
#include "sorter_defs_v1_0.h"


void insertsort(ADDR_t startAddr, LSIZE_t listSize, DATA_t sortList[memSize])
{
	/*-----------------------------------------------------------
	 * Pragmas for startAddr, listSize and the control ports
	 ----------------------------------------------------------*/
	// these pragmas bundle the input arguments into an AXI lite
	// interface for software control
	 #pragma HLS INTERFACE s_axilite port=startAddr   bundle=CTL
	 #pragma HLS INTERFACE s_axilite port=listSize    bundle=CTL
	 #pragma HLS INTERFACE s_axilite port=return      bundle=CTL



	/*-----------------------------------------------------------
	 * Pragmas for the memory interface
	 ----------------------------------------------------------*/
	// this pragma will implement the memory interface as AXI-MM
      #pragma HLS INTERFACE m_axi port=sortList offset=off

	// these pragmas together will implement the interface to the
	// memory as a single port BRAM
     //  #pragma HLS INTERFACE bram port=sortList
     //  #pragma HLS RESOURCE variable=sortList core=RAM_1P


		for (unsigned int i = startAddr+1; i < listSize+startAddr; i++)
		{
			int j;
			DATA_t temp;

		    temp = sortList[i];
		    j = i-1;

		    while (j >= startAddr && sortList[j] > temp)
		    {
		       sortList[j+1] = sortList[j];
		         j = j-1;
		    }
		    sortList[j+1] = temp;
		}


}
```

Header file:

```
#ifndef _SORTER_DEFS_
#define _SORTER_DEFS_

#include <cstdlib>
#include <ctime>

#include <iostream>
#include <iomanip>

using namespace std;



/*-------------------------------------------
  TYPES & CONSTANTS
-------------------------------------------*/

// this defines the total size of the external memory
const unsigned int memSize = 1024;

// this defines the type of the list elements
typedef unsigned int DATA_t;

// this defines the type of the list start address
typedef unsigned int ADDR_t;

// this defines the type of the list size
typedef unsigned int LSIZE_t;


/*-------------------------------------------
  FUNCTIONS PROTOTYPES
-------------------------------------------*/
void insertsort(ADDR_t startAddr, LSIZE_t listSize, DATA_t sortList[memSize]);

void chkResults(DATA_t testArray[memSize], ADDR_t startAddr, LSIZE_t listSize);


#endif
```

Testbench:

```
#include "../src/sorter_defs_v1_0.h"

//#define NDEBUG
#include <assert.h>


int main() {

      // test variables
    ADDR_t tb_startAddr = 13;
    LSIZE_t tb_listSize = 27;
    unsigned int tb_arraysize = 0;


      DATA_t  temp = 0;
      DATA_t tb_array[memSize];
      tb_arraysize = tb_startAddr + tb_listSize;


      // ALL SAME TEST
      cout << "---------------------------------" << endl;
      for (int i=0; i < memSize; i++) {
            tb_array[i] = 7;
      }
    insertsort(tb_startAddr, tb_listSize, tb_array);
    chkResults(tb_array, tb_startAddr, tb_listSize);
      cout << "PASSED all same case" << endl;
      cout << "---------------------------------" << endl;


    // MAX VALUE FIRST
      for (int i=0; i < memSize; i++) {
            tb_array[i] = 7;
      }
      tb_array[tb_startAddr] = 10;
      insertsort(tb_startAddr, tb_listSize, tb_array);
      chkResults(tb_array, tb_startAddr, tb_listSize);
      cout << "PASSED max value first case" << endl;
      cout << "---------------------------------" << endl;


    // MAX VALUE LAST
      for (int i=0; i < memSize; i++) {
            tb_array[i] = 7;
      }
      tb_array[tb_arraysize-1] = 10;
      insertsort(tb_startAddr, tb_listSize, tb_array);
      chkResults(tb_array, tb_startAddr, tb_listSize);
      cout << "PASSED max value last case" << endl;
      cout << "---------------------------------" << endl;


    // MIN VALUE FIRST
      for (int i=0; i < memSize; i++) {
            tb_array[i] = 12;
      }
      tb_array[tb_startAddr] = 4;
      insertsort(tb_startAddr, tb_listSize, tb_array);
      chkResults(tb_array, tb_startAddr, tb_listSize);
      cout << "PASSED min value first case" << endl;
      cout << "---------------------------------" << endl;


    // MIN VALUE LAST
      for (int i=0; i < memSize; i++) {
             tb_array[i] = 13;
      }
      tb_array[tb_arraysize-1] = 10;
      insertsort(tb_startAddr, tb_listSize, tb_array);
      chkResults(tb_array, tb_startAddr, tb_listSize);
      cout << "PASSED min value last case" << endl;
      cout << "---------------------------------" << endl;


      // INCREMENTING NUMBERS TEST
      for (int i=0; i < memSize; i++) {
             tb_array[i] = i;
      }
      insertsort(tb_startAddr, tb_listSize, tb_array);
      chkResults(tb_array, tb_startAddr, tb_listSize);
      cout << "PASSED incrementing numbers case" << endl;
      cout << "---------------------------------" << endl;


      // DECREMENTING NUMBERS TEST
      temp = memSize - 1;
      for (int i=0; i < memSize; i++) {
             tb_array[i] = temp;
             temp--;
      }
      insertsort(tb_startAddr, tb_listSize, tb_array);
      chkResults(tb_array, tb_startAddr, tb_listSize);
      cout << "PASSED decrementing numbers case" << endl;
      cout << "---------------------------------" << endl;



    // RANDOM NUMBERS TEST
      srand((unsigned)time(NULL));

      for (int i=tb_startAddr; i < memSize; i++) {
             DATA_t tb_random = rand();
             tb_array[i] = tb_random;;
      }
      insertsort(tb_startAddr, tb_listSize, tb_array);
      chkResults(tb_array, tb_startAddr, tb_listSize);
      cout << "PASSED random numbers case" << endl;
      cout << "---------------------------------" << endl;


  return 0;

}
```

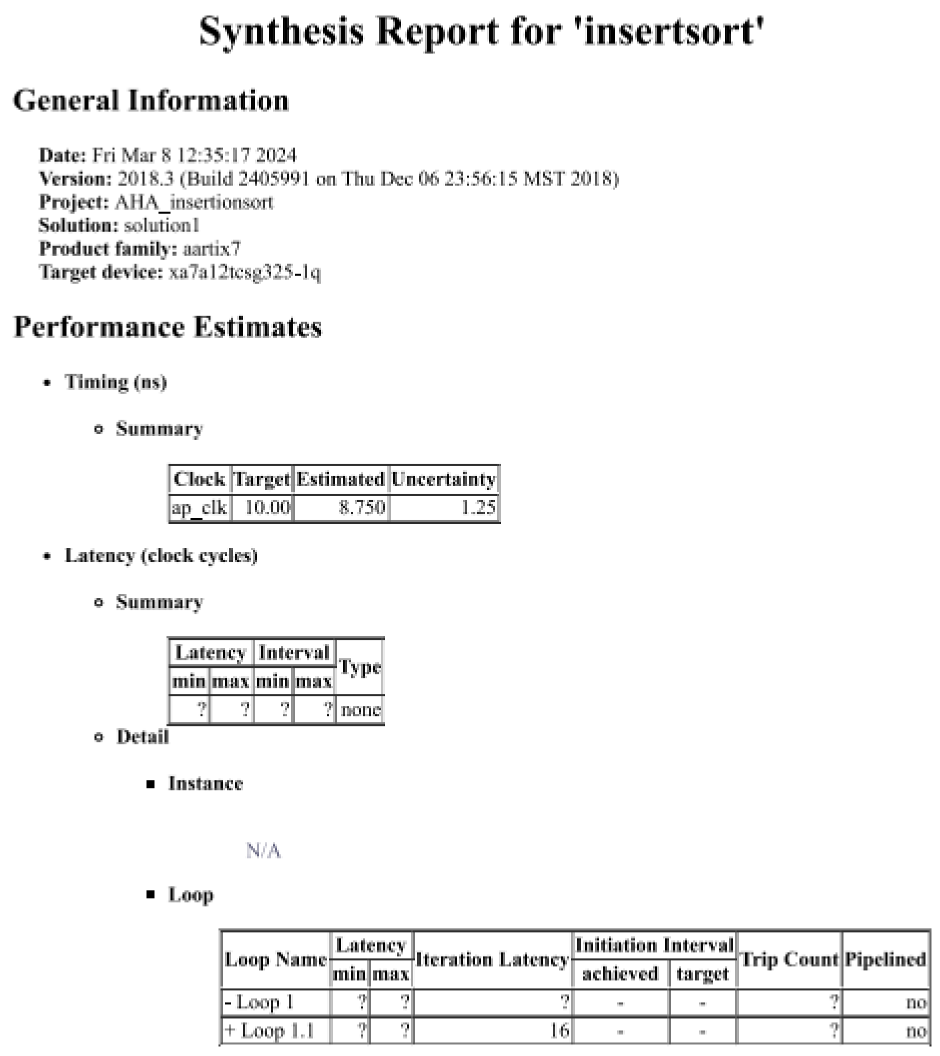
```cpp
void chkResults(DATA_t testArray[memSize], ADDR_t startAddr, LSIZE_t listSize) {

	DATA_t  temp = 0;

	for (int i = startAddr; i < listSize+startAddr; i++) {
		assert (testArray[i] >= temp);
		temp = testArray[i];
		cout << dec << testArray[i] << endl;
	}

}
```

Synthesis Report of HLS Code



Synthesis Report for 'insertsort'

General Information

Date: Fri Mar 8 12:35:17 2024
Version: 2018.3 (Build 2405991 on Thu Dec 06 23:56:15 MST 2018)
Project: AHA_insertionsort
Solution: solution1
Product family: aartix7
Target device: xa7a12tcsg325-1q

Performance Estimates

- **Timing (ns)**
  - **Summary**

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	8.750	1.25

- **Latency (clock cycles)**
  - **Summary**

Latency		Interval		Type
min	max	min	max	
?	?	?	?	none

  - **Detail**
    - **Instance**

N/A

    - **Loop**

Loop Name	Latency		Iteration Latency	Initiation Interval		Trip Count	Pipelined
	min	max		achieved	target		
- Loop 1	?	?	?	-	-	?	no
+ Loop 1.1	?	?	16	-	-	?	no

Utilization Estimates

- **Summary**

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	218
FIFO	-	-	-	-
Instance	2	-	624	748
Memory	-	-	-	-
Multiplexer	-	-	-	288
Register	-	-	325	-
Total	2	0	949	1254
Available	40	40	16000	8000
Utilization (%)	5	0	5	15

- **Detail**
  - **Instance**

Instance	Module	BRAM_18K	DSP48E	FF	LUT
insertsort_CTL_s_axi_U	insertsort_CTL_s_axi	0	0	112	168
insertsort_sortList_m_axi_U	insertsort_sortList_m_axi	2	0	512	580
Total	2	2	0	624	748

  - DSP48

  - **Expression**

Variable Name	Operation	DSP48E	FF	LUT	Bitwidth P0	Bitwidth P1
grp_fu_135_p2	+	0	0	39	32	1
i_fu_151_p2	+	0	0	39	32	1
j_1_fu_192_p2	+	0	0	39	32	2
tmp_fu_145_p2	+	0	0	39	32	32
ap_block_state11_io	and	0	0	8	1	1
tmp_1_fu_157_p2	icmp	0	0	18	32	32
tmp_3_fu_172_p2	icmp	0	0	18	32	32
tmp_5_fu_188_p2	icmp	0	0	18	32	32
Total	8	0	0	218	225	133

  - **Multiplexer**

Name	LUT	Input Size	Bits	Total Bits
ap_NS_fsm	153	34	1	34
ap_sig_ioackin_sortList_ARREADY	9	2	1	2
ap_sig_ioackin_sortList_AWREADY	9	2	1	2
ap_sig_ioackin_sortList_WREADY	9	2	1	2
j1_reg_124	9	2	32	64
j_reg_114	9	2	32	64
sortList_ARADDR	15	3	32	96
sortList_AWADDR	15	3	32	96
sortList_WDATA	15	3	32	96
sortList_blk_n_AR	9	2	1	2
sortList_blk_n_AW	9	2	1	2
sortList_blk_n_B	9	2	1	2
sortList_blk_n_R	9	2	1	2
sortList_blk_n_W	9	2	1	2
Total	288	63	169	466

- Register

Name	FF	LUT	Bits	Const Bits
ap_CS_fsm	33	0	33	0
ap_reg_ioackin_sortList_ARREADY	1	0	1	0
ap_reg_ioackin_sortList_AWREADY	1	0	1	0
ap_reg_ioackin_sortList_WREADY	1	0	1	0
i_reg_231	32	0	32	0
j1_reg_124	32	0	32	0
j_1_reg_271	32	0	32	0
j_reg_114	32	0	32	0
reg_141	32	0	32	0
sortList_addr_1_read_reg_262	32	0	32	0
startAddr_read_reg_220	32	0	32	0
temp_reg_246	32	0	32	0
tmp_3_reg_252	1	0	1	0
tmp_reg_226	32	0	32	0
Total	325	0	325	0

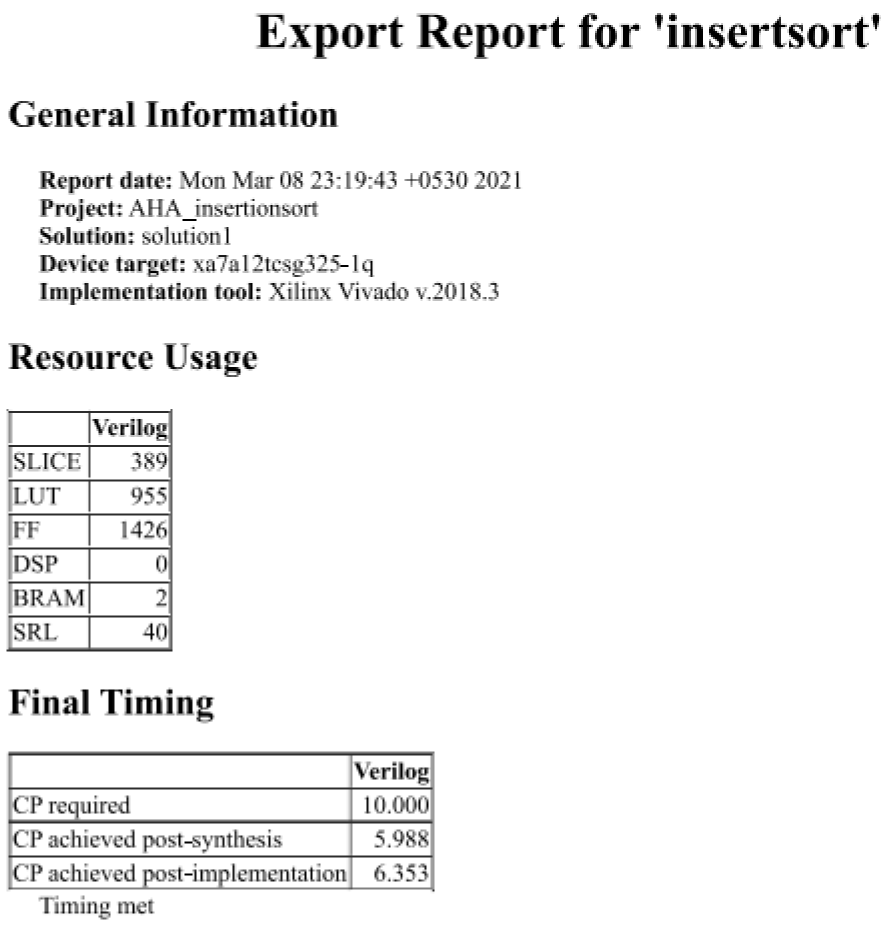
Interface

- Summary

RTL Ports	Dir	Bits	Protocol	Source Object	C Type
s_axi_CTL_AWVALID	in	1	s_axi	CTL	scalar
s_axi_CTL_AWREADY	out	1	s_axi	CTL	scalar
s_axi_CTL_AWADDR	in	5	s_axi	CTL	scalar
s_axi_CTL_WVALID	in	1	s_axi	CTL	scalar
s_axi_CTL_WREADY	out	1	s_axi	CTL	scalar
s_axi_CTL_WDATA	in	32	s_axi	CTL	scalar
s_axi_CTL_WSTRB	in	4	s_axi	CTL	scalar

s_axi_CTL_ARVALID	in	1	s_axi	CTL	scalar
s_axi_CTL_ARREADY	out	1	s_axi	CTL	scalar
s_axi_CTL_ARADDR	in	5	s_axi	CTL	scalar
s_axi_CTL_RVALID	out	1	s_axi	CTL	scalar
s_axi_CTL_RREADY	in	1	s_axi	CTL	scalar
s_axi_CTL_RDATA	out	32	s_axi	CTL	scalar
s_axi_CTL_RRESP	out	2	s_axi	CTL	scalar
s_axi_CTL_BVALID	out	1	s_axi	CTL	scalar
s_axi_CTL_BREADY	in	1	s_axi	CTL	scalar
s_axi_CTL_BRESP	out	2	s_axi	CTL	scalar
ap_clk	in	1	ap_ctrl_hs	insertsort	return value
ap_rst_n	in	1	ap_ctrl_hs	insertsort	return value
interrupt	out	1	ap_ctrl_hs	insertsort	return value
m_axi_sortList_AWVALID	out	1	m_axi	sortList	pointer
m_axi_sortList_AWREADY	in	1	m_axi	sortList	pointer
m_axi_sortList_AWADDR	out	32	m_axi	sortList	pointer
m_axi_sortList_AWID	out	1	m_axi	sortList	pointer
m_axi_sortList_AWLEN	out	8	m_axi	sortList	pointer
m_axi_sortList_AWSIZE	out	3	m_axi	sortList	pointer
m_axi_sortList_AWBURST	out	2	m_axi	sortList	pointer
m_axi_sortList_AWLOCK	out	2	m_axi	sortList	pointer
m_axi_sortList_AWCACHE	out	4	m_axi	sortList	pointer
m_axi_sortList_AWPROT	out	3	m_axi	sortList	pointer
m_axi_sortList_AWQOS	out	4	m_axi	sortList	pointer
m_axi_sortList_AWREGION	out	4	m_axi	sortList	pointer
m_axi_sortList_AWUSER	out	1	m_axi	sortList	pointer
m_axi_sortList_WVALID	out	1	m_axi	sortList	pointer
m_axi_sortList_WREADY	in	1	m_axi	sortList	pointer
m_axi_sortList_WDATA	out	32	m_axi	sortList	pointer
m_axi_sortList_WSTRB	out	4	m_axi	sortList	pointer
m_axi_sortList_WLAST	out	1	m_axi	sortList	pointer
m_axi_sortList_WID	out	1	m_axi	sortList	pointer
m_axi_sortList_WUSER	out	1	m_axi	sortList	pointer
m_axi_sortList_ARVALID	out	1	m_axi	sortList	pointer
m_axi_sortList_ARREADY	in	1	m_axi	sortList	pointer
m_axi_sortList_ARADDR	out	32	m_axi	sortList	pointer
m_axi_sortList_ARID	out	1	m_axi	sortList	pointer
m_axi_sortList_ARLEN	out	8	m_axi	sortList	pointer
m_axi_sortList_ARSIZE	out	3	m_axi	sortList	pointer
m_axi_sortList_ARBURST	out	2	m_axi	sortList	pointer
m_axi_sortList_ARLOCK	out	2	m_axi	sortList	pointer
m_axi_sortList_ARCACHE	out	4	m_axi	sortList	pointer
m_axi_sortList_ARPROT	out	3	m_axi	sortList	pointer
m_axi_sortList_ARQOS	out	4	m_axi	sortList	pointer

Implementation Report Of Exported RTL with place and route



Export Report for 'insertsort'

General Information

Report date: Mon Mar 08 23:19:43 +0530 2021
Project: AHA_insertionsort
Solution: solution1
Device target: xa7a12tcsg325-1q
Implementation tool: Xilinx Vivado v.2018.3

Resource Usage

	Verilog
SLICE	389
LUT	955
FF	1426
DSP	0
BRAM	2
SRL	40

Final Timing

	Verilog
CP required	10.000
CP achieved post-synthesis	5.988
CP achieved post-implementation	6.353

Timing met