NAVAL POSTGRADUATE SCHOOL

Baylor University Engineering & Computer Science

Algorithm and Programming Considerations for Embedded Reconfigurable Computers

Russell Duren,
Associate Professor
Engineering And
Computer Science
Baylor University
Waco, Texas

Douglas Fouts, Professor
Daniel Zulaica, Engineer
Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California

NAVAL POSTGRADUATE SCHOOL

OBJECTIVES

Baylor University Engineering & Computer Science

- Experience and evaluate the learning curve required to become proficient at developing software for the SRC-6e reconfigurable computer.
- Benchmark and compare the performance and correctness of the SRC-6e against computers with traditional architectures.
- Establish libraries of open-source functions.
- Develop a hardware interface between the SRC-6e and a U.S. Navy AN/SPS-65V search radar.
- Develop software for real time processing of radar signals on the SRC-6e.
- Study and experiment with programming languages, methodologies, and environments to move reconfigurable computing closer to the environment most applications developers are already familiar with.

NAVAL POSTGRADUATE SCHOOL

SRC-6e RECONFIGURABLE COMPUTER

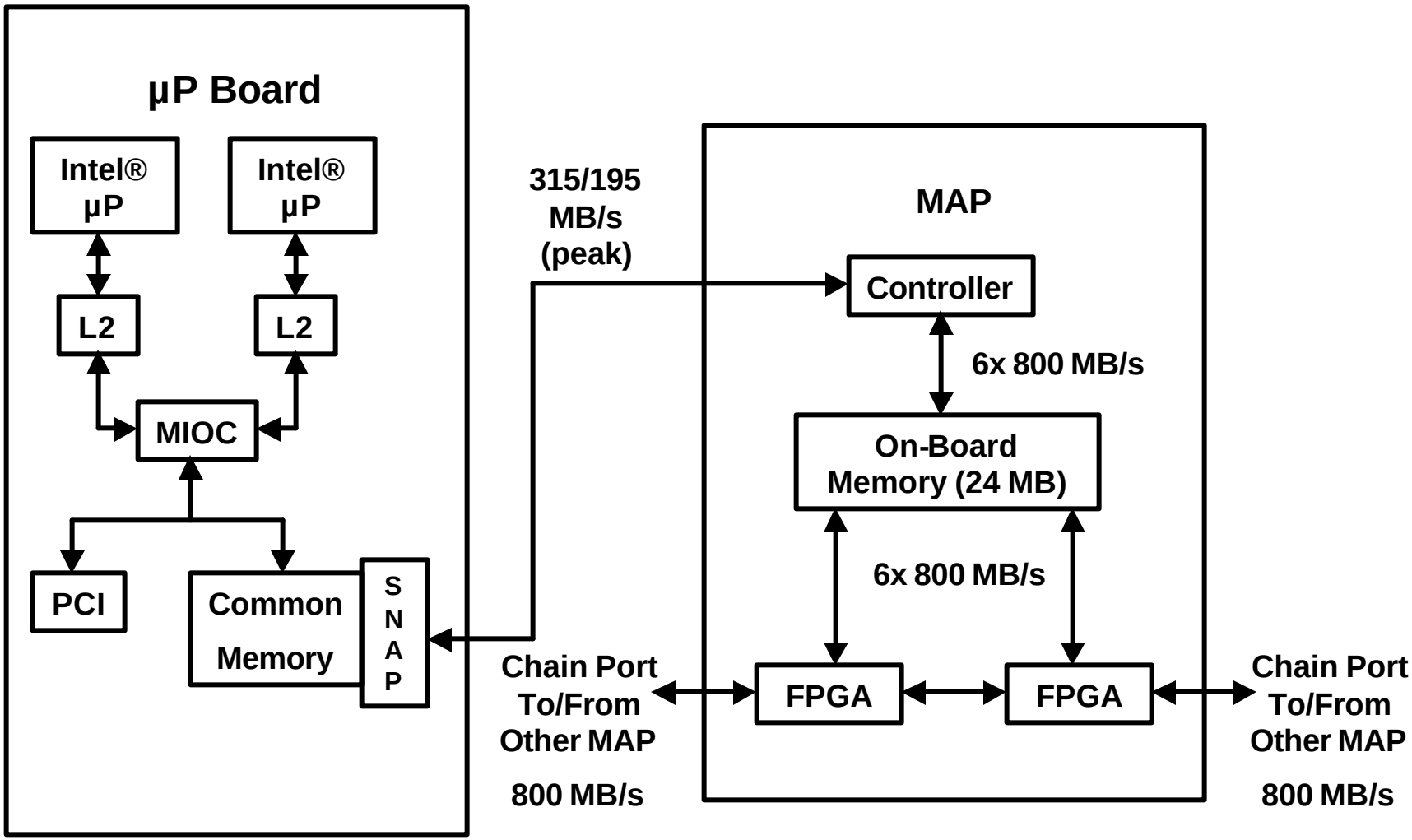
Baylor University Engineering & Computer Science

- LINUX CLUSTER OF TWO PCs
- EACH PC HAS
  - TWO 1000 MHZ INTEL XEON® PROCESSORS
  - COMMON MEMORY
  - SNAP PORT TO MULTI ADAPTIVE PROCESSOR (MAP)
- EACH MAP HAS
  - TWO USER-PROGRAMMABLE XILINX VIRTEX-II FPGAs (6 M GATES EACH)
  - ONE XILINX VIRTEX-II CONTROL FPGA (NOT USER PROGRAMMABLE)
  - ON-BOARD MEMORY
  - SNAP PORT TO PC
  - TWO 96-BIT WIDE CHAIN PORTS TO OTHER MAP
- PROGRAMS WRITTEN IN C OR FORTRAN
  - USER IDENTIFIES WHICH PART(S) OF PROGRAM ARE CONVERTED TO FPGA CIRCUITRY FOR (HOPEFULLY) INCREASED EXECUTION SPEED
  - FPGA CODE CAN ALSO BE WRITTEN IN VHDL OR VERILOG
  - FPGA CAN ALSO BE PROGRAMMED SCHEMATICALLY OR WITH IP CORES

NAVAL POSTGRADUATE SCHOOL

SRC-6E ARCHITECTURE (HALF)

Baylor University Engineering & Computer Science



NAVAL POSTGRADUATE SCHOOL

MAP SOFTWARE DEVELOPMENT

Baylor University Engineering & Computer Science

- CODE FOR FPGAs IS ISOLATED IN EXTERNAL FUNCTION
- SRC COMPILER TRANSLATES C SOURCE CODE INTO VERILOG
- VERILOG IS COMPILED TO FPGA CIRCUITRY
- MAP CAN ALSO BE PROGRAMMED WITH VERILOG, VHDL, IP CORES, OR SCHEMATICALLY
- FPGA CIRCUITRY DEEPLY PIPELINED WITH 100 MHZ CLOCK (10 NS)
- LARGE PIPELINE FILL TIME (LARGE LATENCY)
- CALLS ARE INSERTED IN THE MAIN PROGRAM TO
  - INITIALIZE THE MAP
  - TRANSFER INPUT DATA FROM COMMON MEMORY TO ON-BOARD MEMORY
  - CALL THE EXTERNAL FUNCTION
  - TRANSFER OUTPUT DATA FROM ON-BOARD MEMORY TO COMMON MEMORY
  - RELEASE THE MAP (OPTIONAL)

NAVAL POSTGRADUATE SCHOOL

LIMITATIONS OF MAP C COMPILER

- LIMITED SUBSET OF C LANGUAGE
- LIMITED DATA TYPES (32-BIT AND 64-BIT INTEGERS)
- SNAP PORT DATA TRANSFERS ARE ALWAYS 32-BIT WORDS AND MUST BE ALIGNED ON A 4-BYTE BOUNDARY
- DATA ARRAYS MUST BE ALIGNED ON A 4-BYTE BOUNDARY
- SIX ON-BOARD MEMORY MODULES
- ONE ARRAY PER MEMORY MODULE
- ONE READ OPERATION OR ONE WRITE OPERATION PER MEMORY MODULE PER CLOCK
- NO SUPPORT FOR ACCESS TO CHAIN PORTS

- FUTURE VERSIONS OF MAP C COMPILER WILL FIX SOME, BUT NOT ALL, OF LIMITATIONS

NAVAL POSTGRADUATE SCHOOL

EVALUATING THE SRC-6e AND MAP COMPILER

- EASE OF USE (SOMEWHAT SUBJECTIVE)
  - ALGORITHM/CODE MODIFICATIONS REQUIRED TO MAKE USE OF MAP
  - LIMITATIONS IMPOSED BY COMPILER
  - LIMITATIONS IMPOSED BY HARDWARE
- TYPICAL MEASURES OF COMPILER PERFORMANCE
  - CODE EXECUTION SPEED
  - OBJECT CODE SIZE (TRANSLATES TO FPGA CIRCUIT AREA)

- THE CORDIC ALGORITHM
  - USED TO EXTRACT PHASE INFORMATION FROM AN I/Q-ENCODED (IN-PHASE/QUADRATURE) RADAR SIGNAL
  - USES SUCCESSIVE APPROXIMATION TECHNIQUE TO ESTIMATE ARCTANGENT FUNCTION
  - NUMBER OF ITERATIONS DETERMINES ACCURACY
  - EASILY PIPELINED
  - USES SHIFT AND ADD ALGORITHM, NO MULITPLICATION OR DIVISION

NAVAL POSTGRADUATE SCHOOL

CORDIC TEST CASES

- C VERSION USING ONLY INTEL CPUs
- C VERSION USING SRC C COMPILER FOR MAP
  - 32-BIT INTEGER VARIABLES
  - 11 ITERATIONS
  - MOST CLOSELY MATCHES IP CORE NUMBER 2
- THREE VERSIONS OF MAP CODE GENERATED WITH XILINX IP CORE GENERATOR TOOL
- COMMON “MAIN” PROGRAM
  - CALLS CORDIC ROUTINE 256 TIMES
  - EACH CALL CALCULATES 249984 ARCTANGENTS
- DATA INPUTS TO CORES ARE 8 BITS WIDE
- INTERFACE TO CALLING ROUTINE USES 32-BIT INTEGERS

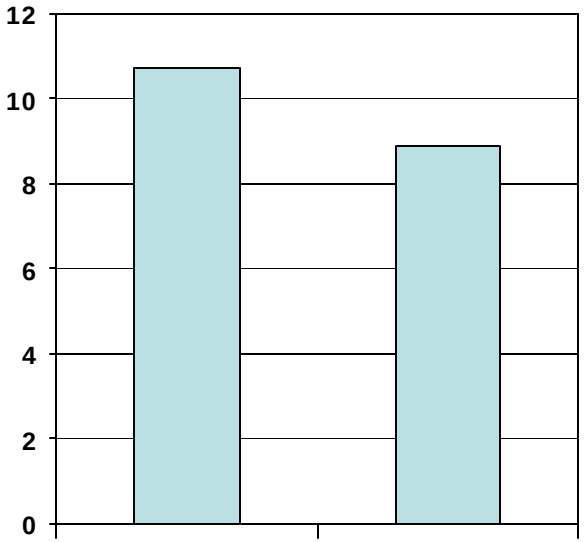
NAVAL POSTGRADUATE SCHOOL

CORDIC EXECUTION TIME

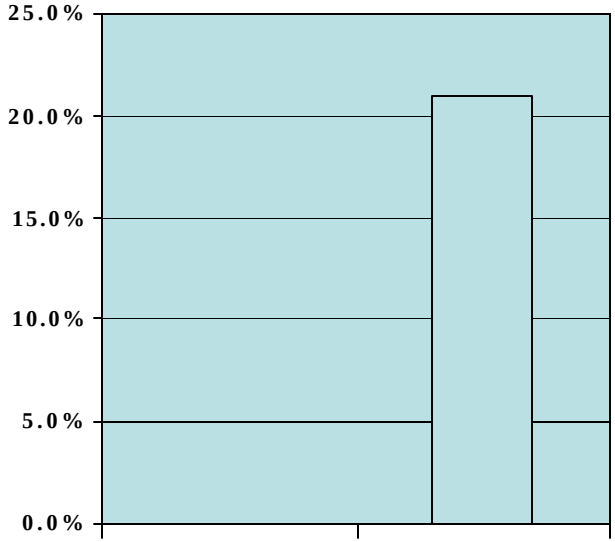
Baylor University Engineering & Computer Science

1000 MHZ INTEL XEON VERSUS 1000 MHZ XEON + MAP

EXECUTION TIME



FPGA SPACE USAGE



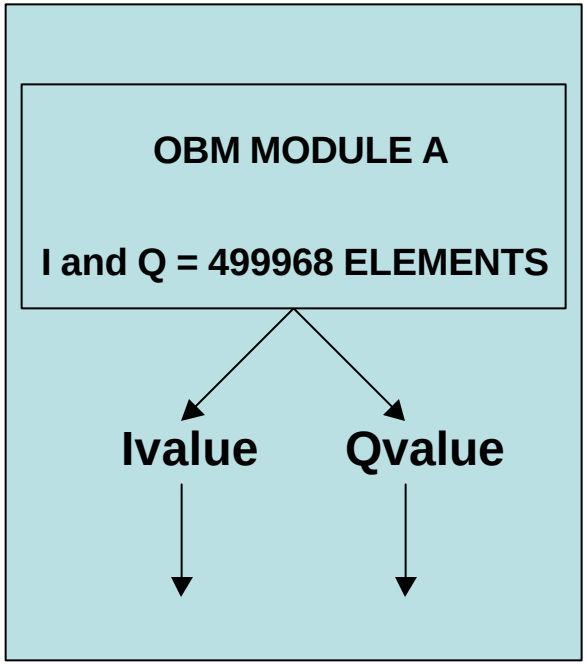
20% DECREASE

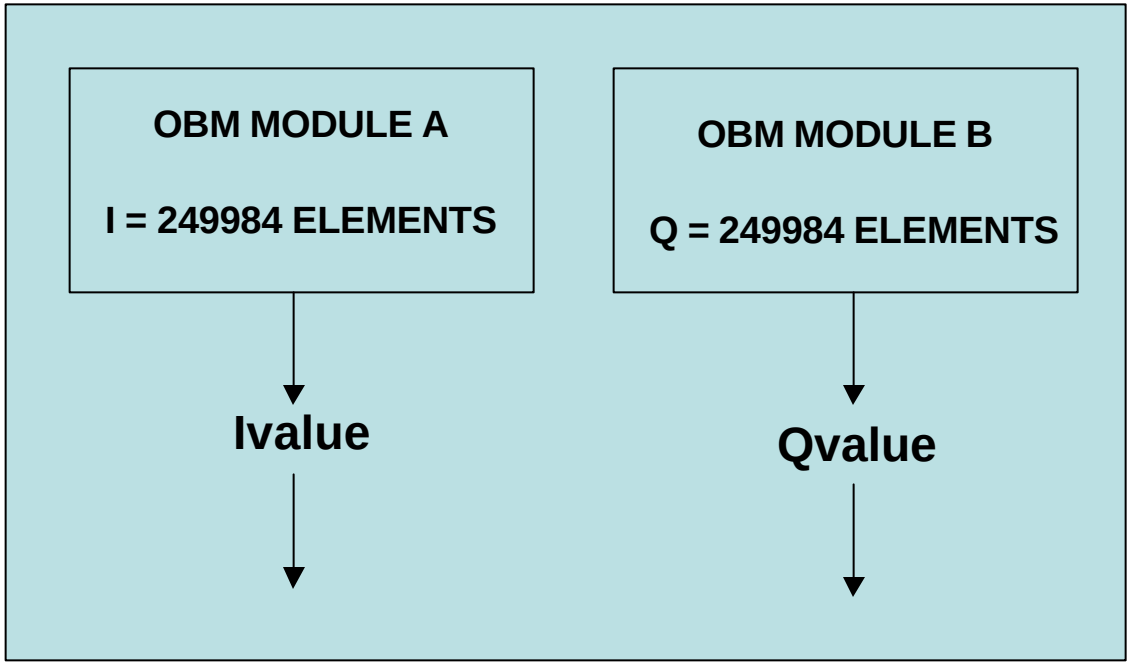
NAVAL POSTGRADUATE SCHOOL

REDUCING MAP DATA ACCESS TIME

Baylor University Engineering & Computer Science

- ORIGINAL MAP CODE STORES I AND Q INPUT DATA IN SINGLE ARRAY
- ARRAYS ALLOCATED TO MODULES IN ON-BOARD MEMORY, BUT ONLY ONE READ OR ONE WRITE OPERATION CAN BE PERFORMED ON EACH CLOCK CYCLE WITH EACH MEMORY MODULE
- USE OF 2 ARRAYS ALLOCATED TO 2 MEMORY MODULES, ONE FOR I AND ONE FOR Q INPUT DATA, REDUCES DATA ACCESS TIME





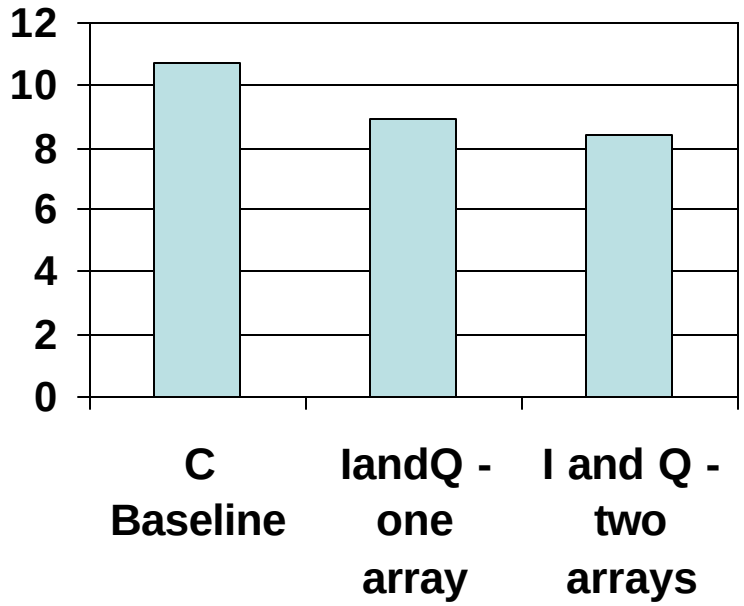
NAVAL POSTGRADUATE SCHOOL

TEST RESULTS

Baylor University Engineering & Computer Science

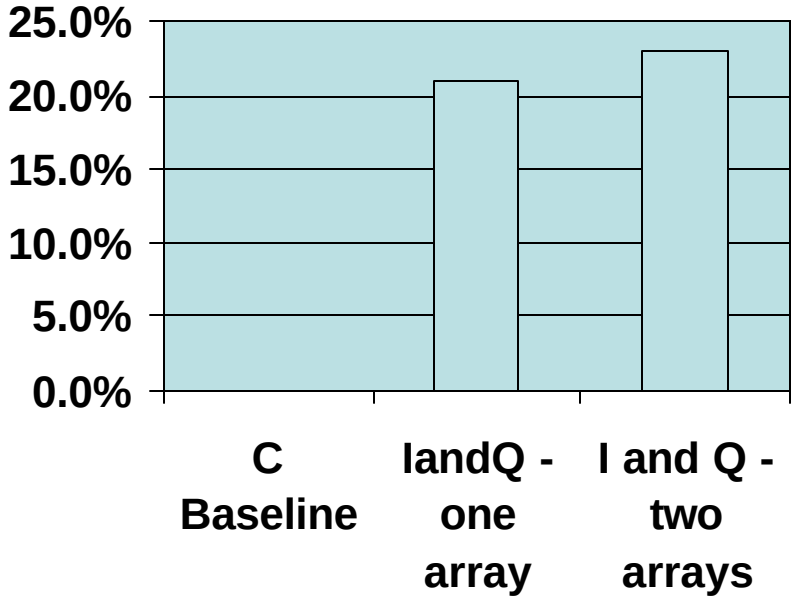
USING SEPARATE ARRAYS TO STORE I AND Q IMPROVES CORDIC EXECUTION TIME AT THE COST OF MORE FPGA SPACE USAGE

EXECUTION TIME



5% DECREASE

FPGA SPACE USAGE



9% INCREASE

NAVAL POSTGRADUATE SCHOOL

REDUCING MEMORY DATA TRANSFER TIME

Baylor University Engineering & Computer Science

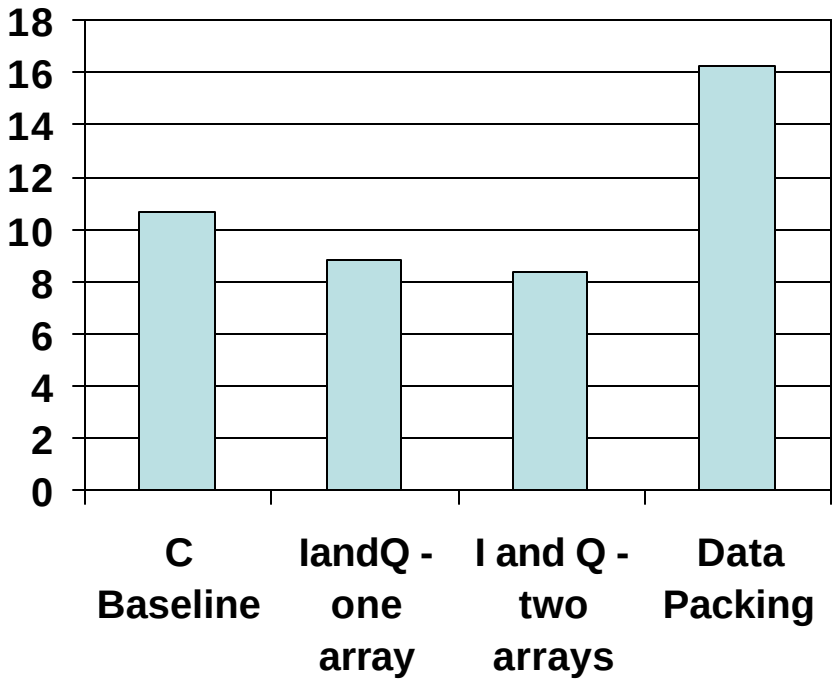
- ORIGINAL MAP CODED ALLOCATED ONE 8-BIT I VALUE AND ONE 8-BIT Q VALUE PER 4-BYTE INPUT WORD AND 1 BYTE OF PHASE DATA PER 4-BYTE OUTPUT WORD
  - DATA TRANSFERS FROM COMMON MEMORY TO ON-BOARD MEMORY TRANSFER 1 UNUSED BYTE FOR EVERY BYTE OF USEFUL INPUT DATA
  - TRANSFERS FROM ON-BOARD MEMORY TO COMMON MEMORY TRANSFER 3 BYTES OF UNUSED DATA FOR EVERY BYTE OF RESULT DATA
- CORDIC WITH DATA PACKING
  - PACK INPUT DATA BEFORE CALLING MAP ROUTINE, 2 I VALUES AND 2 Q VALUES PER 4-BYTE INPUT WORD
  - TRANSFER DATA FROM COMMON MEMORY TO ON-BOARD MEMORY
  - CALL MAP ROUTINE
  - UNPACK DATA INSIDE MAP ROUTINE
  - EXECUTE CORDIC ALGORITHM
  - PACK RESULT DATA, 4 PHASE VALUES PER 4-BYTE OUTPUT WORD
  - TRANSFER DATA FROM ON-BOARD MEMORY TO COMMON MEMORY

NAVAL POSTGRADUATE SCHOOL

TEST RESULTS

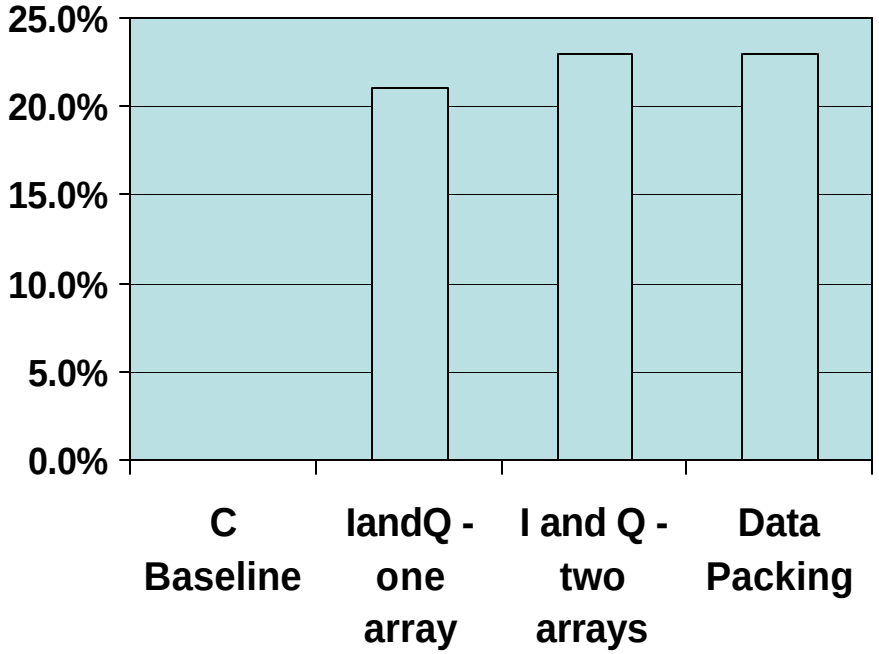
Baylor University Engineering & Computer Science

EXECUTION TIME



190% INCREASE

FPGA SPACE USAGE



INSIGNIFICANT CHANGE

DECREASE IN MEMORY COPY TIME SIGNIFICANTLY LESS THAN TOTAL TIME REQUIRE TO PACK AND UNPACK INPUT AND OUTPUT DATA

NAVAL POSTGRADUATE SCHOOL

MAP CODE GENERATED WITH CORE GENERATOR TOOL

Baylor University Engineering & Computer Science

	PRECISION (BITS)			ITERATIONS	INTERNAL LATENCY	CORE CIRCUIT AREA
	INPUTS	OUTPUT	INTERNAL		(CLOCKS)	(SLICES)
CORE 1	8	9	16	11	17	475
CORE 2	8	9	32	11	17	777
CORE 3	8	9	48	30	36	2604

NAVAL POSTGRADUATE SCHOOL

TEST CASE RESULTS

Baylor University Engineering & Computer Science

	TOTAL CIRCUIT AREA (SLICES*)	TOTAL LATENCY (CLOCKS)	CORDIC EXECUTION TIME (SECONDS)
CORE 1	3260	68	7.3
CORE 2	3555	68	-
CORE 3	5450	81	7.5
C CODE	6710	112	7.4

*33,792 SLICES AVAILABLE

NAVAL POSTGRADUATE SCHOOL

RESULTS ANALYSIS

Baylor University Engineering & Computer Science

- C CODE REQUIRED 2 TO 5 TIMES MORE CIRCUIT AREA
  - 6710 SLICES COMPARED TO 3555 FOR MOST SIMILAR IP CORE
  - APPROXIMATELY 2800 SLICES ADDED TO CORDIC CODE FOR SNAP PORT INTERFACE AND CONTROL
  - 3910 SLICES COMPARED TO 755 SLICES FOR MOST SIMILAR IP CORE
- CIRCUIT AREA LIMITS PROGRAM SIZE
- EXTRA FPGA AREA CAN BE USED FOR LOOP UNROLLING AND PARALLELIZATION TO INCREASE EXECUTION SPEED
- TOTAL EXECUTION TIME IS APPROXIMATELY THE SAME AT 7.4 SEC
- LATENCY FOR C CODE IS ABOUT TWICE THAT OF IP CORES
- LATENCY IS INSIGNIFICANT WITH LARGE DATA BLOCKS
  - Time per loop = (Latency – 1 + Number of Calculations) * 10 ns
  - (112 – 1 + 249984) * 10 ns = 2.5 msec/loop
- CORDIC EXECUTION TIME IS SMALL COMPARED TO TOTAL EXECUTION TIME OF 7.4 SEC
  - 256 loops * 2.5 msec = 0.64 seconds
- MEMORY TRANSFER TIME DOMINATES EXECUTION TIME
  - MEMORY TRANSFER REQUIRE 4 SECONDS AT PEAK TRANSFER RATE

NAVAL POSTGRADUATE SCHOOL

COMPARISON OF MAP PROGRAMMING METHODS

- USE OF MAP C COMPILER
  - LITTLE SPECIAL KNOWLEDGE OF MAP HARDWARE REQUIRED
  - FLEXIBILITY -- PROGRAMMER IMPLEMENTS ALGORITHM AS DESIRED
  - LIMITED TO DATA TYPES RECOGNIZED BY COMPILER
  - LOWEST COST METHOD FOR APPLICATIONS WHERE IP CORES DO NOT EXIST
  - AVOIDS PURCHASE OR ROYALTY COST OF IP CORES WHEN THEY DO EXIST
- USE OF IP CORES
  - REDUCED DEVELOPMENT EFFORT
  - PREVIOUSLY VALIDATED (HOPEFULLY)
  - EASILY RECONFIGURED
  - LOWER LATENCY (PROBABLY NOT SIGNIFICANT)
  - MORE COMPACT CIRCUIT REALIZATION

NAVAL POSTGRADUATE SCHOOL

CONCLUSIONS

Baylor University Engineering & Computer Science

- SRC-6E COMPILER ALLOWS C PROGRAMMERS TO ACCELERATE PROGRAMS WITHOUT BEING CIRCUIT DESIGNERS
- PORTING CODE TO MAP REQUIRES BASIC KNOWLEDGE OF MAP HARDWARE
- MAP C COMPILER PRODUCES CODE THAT COMPARES WELL TO CUSTOM IP CORES
- DEVELOPMENT ENVIRONMENT HAS CAPABILITY TO INTEGRATE IP CORES OR CUSTOM CIRCUIT DESIGNS
- OVERALL PERFORMANCE CAN BE LIMITED BY MEMORY TRANSFER TIME BETWEEN COMMON MEMORY AND ON-BOARD MEMORY
- USE OF LARGE DATA SETS AMORTIZES MAP PIPELINE LATENCY ACROSS MANY CALCULATIONS
- APPLICATIONS PERFORMING A LARGE NUMBER OF CALCULATIONS ON EACH DATA SET DERIVE THE LARGEST PERFORMANCE BOOST FROM USING THE MAP