STAP Radar Processing Design Example

Altera Hardware Implementation



Agenda

Introduction to Altera floating point in FPGAs

- 28 nm silicon enhancements
- "Fused Datapath"

DSP Builder Advanced Blockset overview

STAP Radar processing design example



Floating Point in FPGA - Technology

Hardware	Software & IP
Stratix® V	Fused Data-path Technology
	Patented Algorithms & IP



18x18, 27x27, 36x36 seamless trade-off

Greatly increased multiplier density



© 2010 Altera Corporation—Public



28 nm Variable Precision

Stratix II



Four 18x18 (Independent) Eight 9x9 (Independent) One 36x36 Stratix III / IV, Arria II



Eight 18x18 Multipliers (Sum) Four 18x18 (Independent) Six 12x12 (Independent) Two 36x36 (Independent) Eight 9x9 (Independent) Stratix V



Eight 18x18 (Sum) Four 27x27 High Precision Four 18x36 High Precision Plus more.....

Highest performance, highest precision DSP at 28 nm 2000 GMACs or 1000 GFLOPs in single device

© 2010 Altera Corporation—Public



Variable Precision Stratix V DSP Block





© 2010 Altera Corporation—Public



Support for Floating Point

Both single and double precision





27x27

Double-Precision Mantissa Multiplication



54x54

© 2010 Altera Corporation—Public



Stratix V DSP Block Modes



Floating Point

- (4) SPFP Mantissa Multiplier
- (1) DPFP Mantissa Multiplier

Other Modes

- (4) A *B +/- C
- (4) $(A B)^2 + (C D)^2$

Other Features

- Pre Adder
- Rounder
- Coefficient Banks
- Cascade Adder

Independent Multiplier Modes

- (12) 9x9
- (8) 16x16
- (6) 18x18
- (4) 18x25
- (4) 27x27
- (4) 18x36
- (2) 36x36
- (1) 54x54

MAC Mode (64 bit Acc)

- (4) Sum or Two 18x18 MAC
- (4) 27x27 MAC
- (4) 18x36 MAC (18x25 MAC)

Sum of Multiply

- (4) Sum of Two 18x18
- (2) Sum of Four 18x18
- (2) Sum of Two 18x36
- (2) Sum of Two 27x27

Complex Multiply

- **(2)** 18x18
- **(1) 27x27**
- **(1) 18x36**

© 2010 Altera Corporation—Public



Floating Point Multiplier Capabilities

- Floating point density largely determined by hard multiplier density
 - Multipliers must efficiently support floating point mantissa sizes



Multipliers vs Stratix III / IV / V

© 2010 Altera Corporation—Public



Floating Point Multiplier Capabilities

- Floating point density largely determined by hard multiplier density
 - Multipliers must efficiently support floating point mantissa sizes



Multipliers vs Stratix III / IV / V

© 2010 Altera Corporation—Public



"Fused Datapath" for Floating Point Functions



Floating Point Methodology

- Processors each FP operation has data
 I/O in standardized IEEE754 format
- This can be done but not optimized in FPGAs
 - Excessive logic usage
 - Unsustainable routing requirements
 - Sub 100 MHz performance
 - This penalty discourages use of FP compared to fixed



© 2010 Altera Corporation—Public

New Floating Point Methodology

- Processors each FP operation has data
 I/O in standardized IEEE754 format
- This can be done but not optimized in FPGAs
 - Excessive logic usage
 - Unsustainable routing requirements
 - Sub 100 MHz performance
 - This penalty discourages use of FP compared to fixed
- Altera has novel approach: "Fused Datapath"
 - IEEE754 interface only at algorithm boundaries
 - Large Reduction in logic and routing
 - Optimize algorithms to use hard multipliers
 - Single and Double Precision Floating Point support
 - Based upon internal C to datapath tool



© 2010 Altera Corporation—Public

Optimized "Fused Datapath" Cores

IEEE754 interface only at algorithm boundaries

- Large Reduction in logic and routing
- Optimize algorithms to use hard multipliers

ADD/SUB	EXPONENT	ABS	MATRIX MULT
DIVIDE	INVERSE	COMPARE	MATRIX INVERT
MULTIPLY	LOG	CONVERT	Sine
SQ ROOT	INV SQ ROOT	FFT	Cosine

Largest Portfolio of floating-point cores

© 2010 Altera Corporation—Public



Pick "multiplier intense" algorithms to build floating point functions

Function	ALUTs	Register	Multipliers (27x27)	Latency	Performance
ALU	541	611	n/a	14	497 MHz
Multiplier	150	391	1	11	431 MHz
Divider	254	288	4	14	316 MHz
Inverse	470	683	4	20	401 MHz
SQRT	503	932	n/a	28	478 MHz
Inverse SQRT	435	705	6	26	401 MHz
EXP	626	533	5	17	279 MHz
LOG	1889	1821	2	21	394 MHz

Little difference between add/subtract and common MATH.H functions CPU can have orders of magnitude cost difference: GOPS ≠ GFLOPS Stratix Series FPGAs: GOPS ≈ GFLOPS

© 2010 Altera Corporation—Public



Fused Data-path Technology: Floating Point FFT MegaCore

One FFT 1024pt Core	FPGA Resources	Actual Resources	Percentage
ALUTs	17,668	8,080	31%
Registers	16,068	58,080	28%
Logic utilization	24,104	58,080	42%
Block memory bits	140,340	6,617,088	2%
M9K Blocks	89	462	19%
Multipliers	64	384	17%
Fmax	320.62 MHz		
Fourteen FFT 1024pt Cores			
ALUTs	237,235	424,960	56%
Registers	254,671	424,960	60%

Logic utilization 343,787 424,960 Block memory bits 1,942,304 21,233,664 M9K Blocks 1162 1280 **Multipliers** 896 1024 299.13 MHz

Fmax

81%

9%

91%

88%

"Fused Datapath" integrated into DSP Builder

MATH.H

- SIN
- COS
- TAN
- ASIN
- ACOS
- ATAN
- EXP
- LOG
- SQRT
- DIVIDE
- SQRT



What is DSP Builder ?

© 2010 Altera Corporation—Public



Simulink Model Based Algorithm Development



© 2010 Altera Corporation—Public



Mathworks' Matlab

 Computer language of DSP engineers

Matlab

- Develop algorithms and applications
- Analyze and access data
- Visualize data
- Perform numeric calculations





© 2010 Altera Corporation—Public



Why Simulink ?

 Dynamic graphical modeling environment

Simulink

- Dynamically develop entire systems
- Simulate and interact with the system
- Explore different architectures
- Analyze results

Simulink adds the concept of cycles (clocks)

- Hardware realizable





© 2010 Altera Corporation—Public

Simulink Blockset Libraries

Simulink

- Sources
- Sinks
- Continuous
- Discrete
- Non-linear
- Math
- Altera DSP Builder Advanced Blockset
- Altera DSP Builder Blockset
- Fixed-point blockset
- DSP blockset
- Real-time workshop
- Communications blockset
- Image acquisition toolbox
- Many others...

□ 🚅 -¤ 🦓			
Base Blocks: AionModelIP/Base Blocks			
🖭 🙀 Simulink		Base Blocks	
Altera DSP Builder Advanced Blocks	: ₊ =		
FFT Blockset		FFT Blockset	
- Filters	+		
🔄 ModelBus		Filters	
	+		
Waveform Synthesis		ModelBus	
- B- AltLab		ModelPrim	
🖄 Arithmetic			
Đ 🗠 📴 Boards		Waveform Synthesis	
Complex Type			
Bate & Control			
TO & Bus			
Prototype Blocks			
🖄 Rate Change			
🖄 Simulation Blocks Library			
🖄 State Machine Functions			
12- Storage			
wideo and image Processing			

© 2010 Altera Corporation—Public



DSP Builder Design Flow



© 2010 Altera Corporation—Public

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



- U ×

DSP System Level Design Flow



© 2010 Altera Corporation—Public



DSP Builder System Level Design Flow



© 2010 Altera Corporation—Public



DSP Builder Overview



© 2010 Altera Corporation—Public



DSP Builder Advanced Blockset



© 2010 Altera Corporation—Public

DSP Builder Advanced Blockset

- Constraint-driven behavioral design
 - Set desired clock frequency
- Models silicon speeds when translating to HDL
 - Different devices families / speed grades result in different HDL
- Automated resource sharing
 - Tool analyzes clock rate in relation to sample rate, # of channels, interpolation/decimation factor and develops efficient HDL

Automated pipelining

- To meet desired clock rate
- Enable timing closure at high clock rates of 400-500 MHz

Increased Productivity by Closing Timing Faster

© 2010 Altera Corporation—Public



Unique HDL for Different fmax Requirements



- Simply enter desired System Clock Frequency,
- No need to change model
- Simple 50-bit 4-input adder tree
 - 100 MHz Target => 118 LUT4s, 121 MHz, No pipeline
 - 200 MHz Target => 175 LUT4s, 286MHz, 1 stage pipeline
 - 400 MHz Target => 350 LUT4s, 581 MHz, 5 stage pipeline

Timing driven synthesis produces small or fast RTL from same model

© 2010 Altera Corporation—Public



ModelPrim: Zero Latency Blocks

Blocks are behavioural in nature

- What to do, not When to do it
- Focus on signal flow representation
- Much easier debug and modify without pipeline



Behavioural input enables Optimizations

© 2010 Altera Corporation—Public



DSP Builder Advanced Blockset Feature Overview





© 2010 Altera Corporation—Public

Works Within the Industry's DSP **Development Environment**



30

Allows You to Design Behaviorally

No need for knowledge of silicon features



© 2010 Altera Corporation—Public



Automatically Write HDL to Meet Fmax

Unique HDL written to meet Fmax without any intervention

r(Block Parameters: Signals	
rtsa_frontend_v3/filter *	DSP Builder Advanced Blockset Signals Block (mask) (link)	
	The Signals block specifies information about the dock, reset, and memory bus signals used by the simulation model and the hardware generation	*
	Parameters	Slow 000mV/ 85C Model Emark Summary
	Clock	Slow 900mv 85C Prodel Pinax Summary
		Fmax Restricted Fmax Clock Name Note
2 boolean	Clock Frequency (MHz)	1 409.84 MHz 409.84 MHz clk
<u> </u>	375	
	Clock Margin (MHz)	
1 sflv9_En7 (c) (48) d0	0	22 + 00 00 = = = = 00 (24) + (1) = = (1) = = (1) = = (1) = (
Char	Reset	0) ChannelOut
	areset	2 x stu28_En24 (2 (240) Arrs) stu22_En24 (2) (24) 440 crN
	Reset Active High 👻	Multi Array of Primitive 24 sth/33_En24 (ct_a^2 convert sth/10_En7 (c) (24)
	Bus Name	Addicentertap Convert
	bus	
	Separate Bus Clock	
	Bus Clock Frequency (MHz)	
	ClockRate/4	
	Bus Clock Synchronous with System Clock	
Ready		74% FixedStepDiscrete //
© 2010 Altera Co	<u>QK</u> <u>Cancel</u> <u>H</u> elp <u>A</u> pply	



Efficiently Time Shares Resources

 Automatically time shares resources efficiently when clock rate > sample rate



© 2010 Altera Corporation—Public



Design Space Exploration Across Families

Lets you target different device families without changing any of your design







Block Parameters: Device
DSP Builder Advanced Blockset Device Block (mask) (link)
Marks a particular Simulink Subsystem as the top level of an FPGA device and sets up device options.
Parameters
Device Family Cydone III LS 🔹
Family Member
AUTO
Speed Grade -6
QK <u>C</u> ancel <u>H</u> elp Apply

Block Parameters: Device	
DSP Builder Advanced Blockset Device Block (mask) (link)	D
Marks a particular Simulink Subsystem as the top level of an FPGA device and sets up device options.	M
Parameters	-P
Device Family Stratix IV	D
Device Stratix IV GX 🔹	D
Family Member	F
EP4SGX230HF35I3	
Speed Grade -3	s
OK Cancel Help Apply	

Block Parameters: Device
DSP Builder Advanced Blockset Device Block (mask) (link)
Marks a particular Simulink Subsystem as the top level of an FPGA device and sets up device options.
Parameters
Device Family Stratix V
Device Stratix V GX 🗸
Family Member
AUTO
Speed Grade -3
QK <u>C</u> ancel <u>H</u> elp <u>Apply</u>

© 2010 Altera Corporation—Public



"Future Proof" Your Designs

Lets you target future devices families without changing any of your design



Block Parameters: Device
DSP Builder Advanced Blockset Device Block (mask) (link)
Marks a particular Simulink Subsystem as the top level of an FPGA device and sets up device options.
Parameters
Device Family Stratix VI
Device Stratix VI GX 🔹
Family Member
АЛТО
Speed Grade -3
OK Cancel Help Apply



DSP Builder Ad	vanced Blockset Device Block (mask) (link)
Marks a particu up device optic	lar Simulink Subsystem as the top level of an FPGA device and sets ons.
Parameters	
Device Family	Arria VI GX 🗸
Family Member	
AUTO	
Speed Grade	4

© 2010 Altera Corporation—Public



Obtain Accurate Resources

No Need To Compile....

🚱 Help						
<u>File E</u> dit <u>V</u> iew <u>G</u> o F <u>a</u> vorites	<u>D</u> esktop <u>W</u> indow <u>H</u> elp					
Search 🔎 🔻 🛠	🗰 🌣 🔹 demo_idct8x8/Control					•
Contents Search Results	Help on core functionality of CONTROL					
Release Notes Release Notes MATLAB Communications Toolbox	Resource Utilization					
Embedded MATLAB	Component	Туре	LUT4s	Mults	Memory bits	[
🖶 🔶 Filter Design Toolbox 🛛 🚽	demo_idct8x8	SYNTH	0	0	0	
🕀 🥩 Filter Design HDL Coder	demo_idct8x8/IDCTChip	SYNTH	0	0	0	
H H Image Acquisition Toolbox	demo_idct8x8/IDCTChip/IDCT	SYNTH	0	0	0	
Mage Acquisition Toolbox Mage Processing Toolbox	demo_idct8x8/IDCTChip/IDCT/Col	SYNTH	3550	204	1564	
🐵 蒆 MATLAB Report Generator	demo_idct8x8/IDCTChip/IDCT/CornerTurn	SYNTH	1383	0	703	
🖶 🔶 Optimization Toolbox 🖉 🖉	demo_idct8x8/IDCTChip/IDCT/Row	SYNTH	1695	7	738	
	r					

© 2010 Altera Corporation—Public


Supports Complex Control Logic

Nested and Sequential For Loops



© 2010 Altera Corporation—Public



Fixed and Floating Point in Same Model

Complex as well (c) = complex data path



Works with Vectorized Busses

Single Multiplier in Model = 34 Single Precision Floating Point Complex Multipliers



Direct RF Support

Supports sample rates >> clock rate
Automatically manages multiple phases



© 2010 Altera Corporation—Public



"Fused Datapath" integrated in DSPBuilder

Standard Blocks

0.5 Const Add CmpEQ Abs >a select a>=b AddSLoad >d0 mux Sub CmpGE conj d1 min a<b Mux ComplexConjugate Select Min CmpLT max -= LUT Max CmpNE Mult. SampleDelay Lut sin(a) asin(a) log(a) b Divide Sin ASin Log cos(a) acos(a) exp(a) sqrt(a) Cos ACos Exp RecipSqRt ldexp tan(a) atan(a) LdExp Tan ATan Reciprocal

Math.h Library

© 2010 Altera Corporation—Public



Simplifies Realization of Complex Equations

Black-Scholes Example:

$$C = SN(d) - Le^{-rt}N(d - \sigma\sqrt{t})$$

where the variable d is defined by:



 $d = (ln(S/L) + v^*v^*t/2 + rt)/sqrt(v^*v^*t)$

© 2010 Altera Corporation—Public



Simplifies Realization of Complex Equations

Drop down blocks for operators...



$d = (ln(S/L) + v^*v^*t/2 + rt)/sqrt(v^*v^*t)$

© 2010 Altera Corporation—Public



Memory Synthesis

- Automated processor <> hardware interface synthesis
- Builds pipelined memory mapped interface logic for:
 - IP (eg filter coefficients)
 - Primary Systems (eg registers, shared memories)



© 2010 Altera Corporation—Public



Multi-channel Designs

- IIR example uses 'textbook' lumped delays
- Replace registers with number of channels
- Delays are distributed around logic to meet fmax goal
- Processes multiple channels simultaneously



© 2010 Altera Corporation—Public





Supports Flow Control

Avalon Streaming Interfaces and FIFOs



© 2010 Altera Corporation—Public



SOPC Builder Integration

- Class.ptf files generated for each system
- Slower bus clock connects to system
- Faster signal processing clock is external
- Hierarchical address spaces, so systems can be easily integrated

ڬ Altera SOPC Builder - aion								
File Module System View Tools Help								
System Contents Board Settings Nios II More "cpu_0" Settings System Generation								
Altera SOPC Builder	_ Tar	rget			0 - 0 -			
Create New Component	Boa	ard: Nios Development Board, Strativ II.	(EP2560)	Clock	Source	1Hz Pipe	line	
Avaion Components		ard. Mos bevelopment board, birddx II (Li 2000)	CIK aliala ta add	External 50		4	
Aion Systems ModellP Device Family: Stratix II HardCopy Compatible								
DUCDatapath	Use	Module Name	Description		Input Clock	Base	End	IRQ
 FilterS DUCDatapath Subsy 	7 İ	🖃 сри_0	Nios II Processor	r - Altera Corporation	n clk			
Bridges		instruction_master	Master port					
Installed Versio	n: 1.0	data_master	Master port			IRQ 0	IRQ 31	ل ہے
		itag_debug_module	Slave port			0x00000000	0×000007FF	
All Available Components		► + + + + onchin_memory_0	On-Chip Memory	(RAM or ROM)	clk	0x00001000	0×00001FFF	
		►	DDCTopSystem		clk	0x00004000	0×00007FFF	
		∽⊞ DUCDatapath_0	DUCDatapath		clk	0x00002000	0×00002FFF	
Add 🦃 Check			Move Up	Move D	own			

Build Complete, High Performance Programmable DSP systems

© 2010 Altera Corporation—Public



Documentation Generation

Memory mapped information is collated into XML and processed to HTML for integrated Matlab Help



System level memory map is processed to datasheet representation

© 2010 Altera Corporation—Public



Ideal DSP Algorithm Development Tool



© 2010 Altera Corporation—Public



Constraint Driven Design: (1) Create Model

Use ModelIP or ModelPrim Libraries





© 2010 Altera Corporation—Public



Constraint Driven Design: (2) Select Device

Device independent modeling until this level



© 2010 Altera Corporation—Public



Constraint Driven Design: (3) Set Frequency

Automatic Pipelining / Time Sharing (ModelIP)

🖬 Block Parameters	: Signals			×
- DSP Builder Advanced Blockset Signals Block (mask) (link)				
The Signals block specifies information about the clock, reset, and memory bus signals used by the simulation model and the hardware generation				
Parameters				
Clock				
clk				
Clock Frequency (MH:	z)			
400				
Clock Margin (MHz)				
C				
Reset				
areset				
Reset Active High				•
Bus Name				
bus				
🔲 Separate Bus Cloc	k			
Bus Clock Frequency	(MHz)			
400				
🗖 Bus Clock Synchro	nous with Syste	m Clock		
	ΠΚ	Cancel	Help (Applu
	21			CPPPy

© 2010 Altera Corporation—Public



Constraint Driven Design: (4) Compile

Compilation Report 🖹 Legal Notice Flow Summary 🧮 Flow Settinas 📕 Flow Non-Default Global Settings Flow Elapsed Time Flow Log Analysis & Synthesis + Fitter + Assembler + TimeQuest Timing Analyzer -🔲 Summary I 📕 SDC File List 📕 Clocks Slow 900mV 85C Model 🗃 🎹 Fmax Summary

Sŀ	Slow 900m¥ 85C Model Fmax Summary						
	Fmax	Restricted Fmax	Clock Name	Note			
1	307.22 MHz	307.22 MHz	bus_clk				
2	408.66 MHz	408.66 MHz	clk 🗲		_		
itter Status Successful - Eri Apr 11 11:35:26 2008							

8.0 Internal Build 185 03/20/2008 SJ Full Version Quartus II Version **Bevision Name** FilterSystem Top-level Entity Name demo_firi_FilterSystem Family Stratix IV Device. EP4SGX70DF29C2 Timing Models Preliminary Logic utilization 7% 🗲 🗕 1,448 / 56,320 (3%) Combinational ALUTs 770 / 28,160 (3%) Memory ALUTs Dedicated logic registers 3,825 / 56,320 (7%) Total registers 3825 Total pins 121 / 412 (29%) Total virtual pins 0 Total block memory bits 12,805 / 6,617,088 (< 1 %) 🔺 14 / 384 (4 %) 🗲 DSP block 18-bit elements Total GXB Receiver Channels 0/16(0%) Total GXB Transmitter Channels 0/16(0%) Total PLLs 0/3(0%) Total DLLs 0/4(0%)

© 2010 Altera Corporation—Public



Higher Level Synthesis



- 1. Covert the MDL schematic into an intermediate DFG representation
- 2. Apply transforms and analysis:
 - Break apart carry chains
 - DSP Block & Memory Timing
 - Share multipliers
 - Pipeline for:
 - required FMax performance
 - Balanced/matched delays

3. Generate RTL

. . . .

© 2010 Altera Corporation—Public

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Offend; and Altera marks in and outside the U.S.



architecture BEHAVIOUR of DSPA is

begin

A := B * C + D;

STAP Radar Design

Floating Point Matrix Operations in FPGAs



© 2010 Altera Corporation—Public

STAP Processing the Radar Cube



© 2010 Altera Corporation—Public



FPGA Processing Flow Implementation (voltage domain) Compute each PRF over L_s range bins $PRF = 1 KH_7$ Compute each target Doppler Vector **F**_d vector, for each PRF Steering Vector t Complex, Single [16 x 1] **Kronecker Precision Floating** Product [192 x 1] Angle Vector $\mathbf{A}_{\mathbf{A}}$ Point Datapath [12 x 1] **y** (range bin **R**₁^H (lower Dot of interest) Forward



出い

© 2010 Altera Corporation—Public

STAP Input Test Data





© 2010 Altera Corporation—Public



Steering Vector Generation

Radar described in m code

🖻 Editor				
<u>File E</u> dit <u>T</u> ext <u>G</u> o <u>C</u> ell T <u>o</u> ols De <u>b</u> ug <u>D</u> esktop <u>W</u> indow <u>H</u> elp		الات ا		
🎦 🚰 📓 👗 ங 🖺 🤊 🝽 🍓 🖅 - 🛤 🖛 🗰 🍂 🕨 - 🔂 🗶 🖷 🎕 🗊 💵 🏭 Stack: Base 🔽 🍂				
. + ↓ ↓ × 1.1				
D:\depot\vmauer_s]-perforce3_1666\depot\ssg\military\radar\STAP\reference\stapSMI.m				
(1) This file uses Cell Mode. For information, see the rapid code iteration video, the publishing video, or help.		×		
61 %% Calculate Weights				
62 % Doppler steering vector				
<pre>63 - dopplerSteeringVec = exp(1i*2*pi*(0:datadim(3)-1)'*dp/prf);</pre>				
64 % Angle steering vector				
65 - nind = single((-(NumElements-1)/2:(NumElements-1)/2).'*(ElementSpacing/lam	nbda));			
66 - angle = single(angle); azang = angle(1,:); elang = angle(2,:);				
67 - angleSteeringVec = exn(-1i*2*ni*(nind*-(cosd(elang).*sind(azang)))):				
68 % Kronecker tensor product				
<pre>69 - stSteeringVec = kron(dopplerSteeringVec, angleSteeringVec);</pre>				
70				
71 % Weights				
72 - [~, R] = qr(TrainingData',0); % R is upper triangular				
<pre>73 - temp = R\(R'\stSteeringVec); % Forward substitution followed by backsubst</pre>	itution			
74 - w = temp/(stSteeringVec'*temp);				
76 %% Calculate output				
$y = w^* \text{ datavec};$				
70 y - y(;);		~		
stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m ×				
	stapSMI Ln 70 Col	1 OVR		

© 2010 Altera Corporation—Public



Steering Vector Generation Port



© 2010 Altera Corporation—Public



Steering Vector Results

Difference- 115dB





Steering Vector Generation

- System requirements
 - Calculate 64 vectors per ms
- Quartus results:
 - 163 18x18,
 - 16K registers, 19K ALUTs,
 - 129 MemLUTs
 - 7 KBit memory
 - 246 MHz (using seed sweep)
 - => Processing time: 500 cycles @ 248 MHz = 2 us
 - Comment: We overachieve by a factor of 500. Timesharing would allow reducing the hardware resources

© 2010 Altera Corporation—Public





QR Decomposition

Radar described in m code

🖻 Editor					
<u>F</u> ile <u>E</u> dit	<u>T</u> ext <u>Go</u> <u>C</u> ell T <u>o</u> ols De <u>b</u> ug <u>D</u> esktop <u>W</u> indow <u>H</u> elp		اد ا		
: 🗋 🖨) 🖆 🔜 🐇 ங 🛍 🤊 (*) 🌭 🖅 - 🛤 🖛 🛶 🎪 🕨 - 🛃 🐮 🖷 🏝 🗊 💷 stack: Base 🕑 🍂				
· • 🖪 📮	-1.0 + \div 1.1 × 9% 9% 0				
D:\depot	\vmayer_si-perforce3_1666\depot\sso\military\radar\STAP\reference\stapSMI_m				
This file	uses Cell Mode. For information, see the rank orde iteration video, the publishing video, or bein				
	assisten model non monitority see the <u>rapid code iteration</u> video, the <u>publishing</u> video, or <u>help</u> .		×		
61	%% Calculate Weights		<u> </u>		
62	% Doppler steering vector				
63 -	dopplerSteeringVec = exp(1i*2*pi*(0:datadim(3)-1)'*dp/prf);				
64	% Angle steering vector				
65 -	nind = single((-(NumElements-1)/2:(NumElements-1)/2).'*(ElementSpacing/lamk	da));			
66 -	angle = single(angle); azang = angle(1,:); elang = angle(2,:);				
67 -	angleSteeringVec = exp(-1i*2*pi*(nind*-(cosd(elang).*sind(azang))));				
68	% Kronecker tensor product				
69 -	stSteeringVec = kron(dopplerSteeringVec, angleSteeringVec);				
70					
71	% Weights				
72 -	[~, R] = qr(TrainingData',O); % R is upper triangular				
73 -	<pre>temp = R\(R'\stSteeringVec); % Forward substitution followed by backsubsti</pre>	tution			
74 -	<pre>w = temp/(stSteeringVec'*temp);</pre>				
75					
76	%% Calculate output		=		
77 -	y = w'*datavec;				
78 -	-y = y(:);		~		
stapSMI.	stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m ×				
		stapSMI Ln	70 Col 1 OVR 🛒		
LTERA, AI	RRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & TI	n. Off.			

QR Decomposition Algorithm Analysis

```
for k=1:n
    r(k,k) = norm(A(1:m, k));
    for j = k+1:n
        r(k, j) = dot(A(1:m, k), A(1:m, j)) / r(k,k);
    end
    q(1:m, k) = A(1:m, k) / r(k,k);
    for j = k+1:n
            A(1:m, j) = A(1:m, j) - r(k, j) * q(1:m, k);
    end
end
```

- Standard algorithm, source: Numerical Recipes in C
- Possible to implement as is, but some changes make it more FPGA friendly and increase numerical accuracy and stability

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



© 2010 Altera Corporation—Public

Parallelism

- Algorithmic requirements
 - m*(k²+k) cmults per QRD
 - $200 * (192^2 + 192) = 7.411.200$
- System requirements
 - Update rate 1 ms:
- Parallelism
 - 7.411.200 cmults / (1ms * 250 MHz) \rightarrow 29.6 cmults in parallel are needed
 - Since matrix is 200: 200/32 = 6.25 clocks per sample = 7 clocks per sample
 - Our design uses 6 clocks per sample \rightarrow 204/6 = 34 parallel paths
- Quartus results:
 - 550 18x18,
 - 120K registers,
 - 96K ALUTs,
 - 8MBit memory

© 2010 Altera Corporation—Public



Structure



© 2010 Altera Corporation—Public



QRD



© 2010 Altera Corporation—Public



R Matrix Results

Is the error too high ? Use double precision









Forward and Back Substitution

Radar described in m code

File Edit Iont © Cell Tools Debug Desktop Window Help Pidepoil Vinsuer_siperforce1_1666/depoilssg/military/radar/STAP/reference/stap5MLm Pidepoil/vinsuer_siperforce1_1666/depoilssg/military/radar/STAP/reference/stap5MLm Pidepoil/vinsuer_siperforce1_1666/depoilssg/military/radar/STAP/reference/stap5MLm Pidepoil/vinsuer_siperforce1_1666/depoilssg/military/radar/STAP/reference/stap5MLm Pidepoil/vinsuer_siperforce1_1666/depoilssg/military/radar/STAP/reference/stap5MLm Pidepoil/vinsuer_siperforce1_1666/depoilssg/military/radar/STAP/reference/stap5MLm Pidepoil/vinsuer_siperforce1_1666/depoilssg/military/radar/STAP/reference/stap5MLm ************************************	🖻 Editor			
Pridepolitymauer_sj.perforce3_1666/depolitsg/military/radar/STAP/reference/stapSMLm <pre></pre>	<u>F</u> ile <u>E</u> dit	<u>T</u> ext <u>Go</u> <u>C</u> ell T <u>o</u> ols De <u>b</u> ug <u>D</u> esktop <u>W</u> indow <u>H</u> elp		لا ا
D:\depol\vmauer_5j-perforce3_1666\depol\ssg\military\radar\STAP\reference\stapSMLm D:\depol\vmauer_5j-perforce3_1666\depol\ssg\military\radar\STAP\reference\stapSMLm @ The file uses Cell Mode. For information, see the read code iteration video, or help. **	: 🛍 🖨	🖆 🔜 👗 🐂 🛍 🤊 🝽 🍓 🖅 - 🛤 🖛 🗰 🎊 🕨 - 🗟 紀 🖷 🕋 🗊 🚛 🏭 Stack: Base 🔽 🍂		
D: depoil Vmauer_sj-perforce 3_1666 \depoilssg\militarylradar\STAP\reference\stapSMLm I * × @ This file uses Cel Mode. For information, see the rapid code terration video, or help. × @ This file uses Cel Mode. For information, see the rapid code terration video, or help. × @ This file uses Cel Mode. For information, see the rapid code terration video, or help. × @ This file uses Cel Mode. For information, see the rapid code terration video, or help. × @ This file uses Cel Mode. For information, see the rapid code terration video, or help. × @ This file uses Cel Mode. For information, see the rapid code terration video, or help. × @ This file uses Cel Mode. For information, see the rapid code terration video, or help. × @ This file uses Cel Mode. For information, see the rapid code terration video, or help. × @ This file uses Cel Mode. For information, see the rapid code terration video, or help. × @ This file uses Cel Mode. For information, see the rapid code terration video, or help. × 65 - nind = single((-(NumElements-1)/2: (NumElements-1)/2).'*(ElementSpacing/lambda)); * 66 - angle = single(angle); acang = angle(1;); elang = angle(2,:); * 70 - angleSteeringVec = ktron(dopplerSteeringVec); angleSteeringVec); * 9 - stSteeringVec = ktron(dopplerSteeringVec); * Forward substitution followed by backsubstitutio	: + <mark>=</mark> _=	$ _{1,0} = _{1,0} + _{1,1} \times _{\infty} \otimes _{\infty} \otimes _{1,0}$		
Winstne dess Cell Mode. For information, see the rapid code iteration video, of help. × 61 % Calculate Weights 62 % Doppler steering vector 63 - dopplerSteeringVec = exp(1i*2*pi*(0:datadim(3)-1)**dp/prf); 64 % Angle steering vector 65 - nind = single((-(NumElements-1)/2).'*(ElementSpacing/lambda)); 66 - angle = single(angle); azang = angle(1,:); elang = angle(2,:); 67 - angleSteeringVec = exp(-1i*2*pi*(nind*-(cosd(elang).*sind(azang)))); 68 % Kronecker tensor product 69 stSteeringVec = kron(dopplerSteeringVec, angleSteeringVec); 70 1 % Weights 72 [~, R] = qr(TrainingData',0); % R is upper triangular 73 temp/(stSteeringVec); % Forward substitution followed by backsubstitution 74 % Calculate output 77 y = w'*datavec; 78 - *** temp/(stSteeringVec)**temp); ** *** stapSMLm × stapSMLm ×	D:\depot	\vmauer_sj-perforce3_1666\depot\ssg\military\radar\STAP\reference\stapSMI.m		□ # X
<pre>00 01 %* Calculate Weights 62 * Doppler steering vector 63 - dopplerSteeringVec = exp(1i*2*pi*(0:datadim(3)-1)'*dp/prf); 64 * Angle steering vector 65 - nind = single((-(NumElements-1)/2).'*(ElementSpacing/lambda)); 66 - angle = single(angle); azang = angle(1,:); elang = angle(2,:); 67 - angleSteeringVec = exp(-1i*2*pi*(nind*-(cosd(elang).*sind(azang)))); 68 * Kronecker tensor product 69 - stSteeringVec = kron(dopplerSteeringVec, angleSteeringVec); 71 * Weights 72 - [~, R] = qr(TrainingData',0); * R is upper triangular 73 - temp = R\(R'\stSteeringVec'* temp); 74 - w = temp/(stSteeringVec'* temp); 75 - 76 ** Calculate output 77 - y = w'*datavec; 78 - y = y(:); 54 stspSML.m × stapSML.m × stapSML.m × 54 stapSML.m × stapSML.m × stapSML.m × 54 stapSML Ln 70 Col 1 07R ;; 55 stapSML Ln 70 Col 1 Col 1 Col 1 07R ;; 55 stapSML Ln 70 Col 1 Col</pre>	This file	uses Cell Mode. For information, see the <u>rapid code iteration</u> video, the <u>publishing</u> video, or <u>help</u> .		X
<pre>0.1 0.5 Calculate stepsing vector 62</pre>	61	22 Calculate Weights		
<pre>da = dopplerSteringVector dopplerSteringVector da = dopplerSteringVector est = nind = single((-(NumElements-1)/2).'*(ElementSpacing/lambda)); ef = angle = single(angle); azang = angle(1,:); elang = angle(2,:); ef = angleSteringVec = exp(-li*2*pi*(nind*-(cosd(elang).*sind(azang)))); ef = angleSteringVec = exp(-li*2*pi*(nind*-(cosd(elang).*sind(azang)))); ef = stSteringVec = kron(dopplerSteringVec, angleSteringVec); ef = stSteringVec = kron(dopplerSteringVec, angleSteringVec); ef = stSteringVec = kron(dopplerSteringVec, angleSteringVec); ef = c, r, R] = qr(TrainingData',0); * R is upper triangular et = temp = R(r\stSteringVec); * Forward substitution followed by backsubstitution ef = temp = (k'\stSteringVec'*temp); e = temp/(stSteringVec'*temp); e = v * datavec; e = y = y(:); e = stapSMLm x stapsMLm x main.m x stapSMLm x etapSML Ln 70 Col 1 OVR ;;</pre>	62	b) calculate weights b) Doppler steering vector		
<pre>do doprint stering vector charactering vector f5 - nind = single((-(NumElements-1)/2:(NumElements-1)/2).'*(ElementSpacing/lambda)); 66 - angle = single(angle); azang = angle(1,:); elang = angle(2,:); 67 - angleSteeringVec = exp(-li*2*pi*(nind*-(cosd(elang).*sind(azang)))); 68 * Kronecker tensor product 69 - stSteeringVec = kron(dopplerSteeringVec, angleSteeringVec); 70 71 * Weights 72 - [~, R] = qr(TrainingData',0); * R is upper triangular 73 - temp = R\[*]stSteeringVec); * Forward substitution followed by backsubstitution 74 - w = temp/(stSteeringVec); * Forward substitution followed by backsubstitution 75 * % Calculate output 77 - y = w'*datavec; 78 - y = y(:); 78 - y = y(:);</pre>	63 -	<pre>> Doppier Sceering Vector dopplerSteeringVect = evp(1)\$2\$pi\$(0.detedim(3)_1)'\$dp/prf).</pre>		
<pre>of _ find = single (-(NumElements-1)/2).'*(ElementSpacing/lambda)); f6f = angle = single(angle); azang = angle(1,:); elang = angle(2,:); f7 = angleSteeringVec = exp(-1i*2*pi*(nind*-(cosd(elang).*sind(azang)))); f8</pre>	64	<pre>> ingle steering vector</pre>		
<pre>And Single(((numbered a))) ((numbered a))) ((numbered a))) ((numbered a))) ((numbered a))) ((numbered a))) ((numbered a)) ((numbered a))) ((numbered a)) ((numbered a)) ((numbered a))) ((numbered a)) ((numbered a)) ((numbered a)) ((numbered a))) ((numbered a)) ((numbered</pre>	65 -	<pre>> Angle Steering vector nind = single((-(NumFlements-1)/2*(NumFlements-1)/2) '*(FlementSnacing/lemin)</pre>	de)).	
<pre>angle teringVec = exp(-li*2*pi*(nind*-(cosd(elang).*sind(azang)))); % Kronecker tensor product 9 - stSteeringVec = kron(dopplerSteeringVec, angleSteeringVec); 0 1 * Weights 72 - [~, R] = qr(TrainingData',0); * R is upper triangular 73 - temp = R\(R'\stSteeringVec); * Forward substitution followed by backsubstitution 74 - w = temp/(stSteeringVec); * Forward substitution followed by backsubstitution 75 - y = w'*datavec; 78 - y = y(:); 53 - y = y(:); 54 - y = y(:); 55 - y = y(:); 57 - y = w'*datavec; 78 - y = y(:); 57 - y = w'*datavec; 79 - y = w'*datavec; 70 - y = w'*datavec; 71 - y = w'*datavec; 72 - y = y(:); 73 - y = y(:); 74 - y = w'*datavec; 75 - y = y(:); 75 - y = y(:); 76 - y = y(:); 77 - y = w'*datavec; 78 - y = y(:); 78 - y = y(:); 79 - y = w'*datavec; 70 - y = w'*datavec; 70 - y = w'*datavec; 71 - y = w'*datavec; 72 - y = w'*datavec; 73 - y = y(:); 74 - y = w'*datavec; 75 - y = y(:); 75 - y = y(:); 76 - y = y(:); 77 - y = w'*datavec; 78 - y = y(:); 78 - y = y(:); 79 - y = w'*datavec; 70 - y = w'*datavec; 70 - y = w'*datavec; 71 - y = w'*datavec; 72 - y = y(:); 73 - y = y(:); 74 - y = w'*datavec; 75 - y = y(:); 75 - y = y(:); 76 - y = y(:); 77 - y = w'*datavec; 78 - y = y(:); 78 - y = y(:); 79 - y = y(:); 70 - y = w'*datavec; 70 - y = y(:); 70 - y = w'*datavec; 70 - y = y(:); 70 - y = y = y = y(:); 70 - y = y = y = y = y = y = y = y = y = y</pre>	66 -	angle = single(angle): azang = angle(1,:): elang = angle(2,:):	saa,,,,	
<pre>ampletecturing too chy(is p,r (num (coold(trang)) cha(doing)))) K Kronecker tensor product stSteeringVec = kron(dopplerSteeringVec, angleSteeringVec); K Weights 2 - [~, R] = qr(TrainingData',0); % R is upper triangular 3 - temp = R\(R'\stSteeringVec); % Forward substitution followed by backsubstitution 4 - w = temp/(stSteeringVec'*temp); K Calculate output 77 - y = w'*datavec; 78 - y = y(:); StapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m × </pre>	67 -	angle Single(angle), asang angle(1,.), clang angle(s,.), angleSteeringVec = exn(-li*2*ni*(nind*-(cosd(elang).*sind(azang)))):		
<pre>69 - stSteeringVec = kron(dopplerSteeringVec, angleSteeringVec); 70 71 * Weights 72 - [~, R] = qr(TrainingData',0); * R is upper triangular 73 - temp = R\(R'\stSteeringVec); * Forward substitution followed by backsubstitution 74 - w = temp/(stSteeringVec'*temp); 75 76 ** Calculate output 77 - y = w'*datavec; 78 - y = y(:); ************************************</pre>	68	* Kronecker tensor product		
70 71 % Weights 72 - [~, R] = qr(TrainingData',0); % R is upper triangular 73 - temp = R\(R'\stSteeringVec); % Forward substitution followed by backsubstitution 74 - w = temp/(stSteeringVec'*temp); 75 75 76 %% Calculate output 77 - y = w'*datavec; 78 - y = y(:); 5 stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m × 5 stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m ×	69 -	stSteeringVec = kron(donnlerSteeringVec, angleSteeringVec):		
<pre>71 * Weights 72 - [~, R] = qr(TrainingData',0); * R is upper triangular 73 - temp = R\(R'\stSteeringVec); * Forward substitution followed by backsubstitution 74 - w = temp/(stSteeringVec'*temp); 75 75 76 ** Calculate output 77 - y = w'*datavec; 78 - y = y(:); 5 stapSMI.m × staplgorithmdemo.m × main.m × stapSMI.m × 5 stapSMI.m × staplgorithmdemo.m × main.m × stapSMI.m × s</pre>	70			
<pre>72 - [~, R] = qr(TrainingData', 0); % R is upper triangular 73 - temp = R\(R'\stSteeringVec); % Forward substitution followed by backsubstitution 74 - w = temp/(stSteeringVec'*temp); 75 76 %% Calculate output 77 - y = w'*datavec; 78 - y = y(:); 78 stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m × stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m ×</pre>	71	* Weights		
73 - temp = R\(R'\stSteeringVec); % Forward substitution followed by backsubstitution 74 - w = temp/(stSteeringVec'*temp); 75 ** Calculate output 77 - y = w'*datavec; 78 - y = y(:); stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m × stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m ×	72 -	[~, R] = gr(TrainingData',O); % R is upper triangular		
74 - w = temp/(stSteeringVec'*temp); 75 %% Calculate output 76 %% Calculate output 77 - y = w'*datavec; 78 - y = y(:); * stapSMLm × stapalgorithmdemo.m × main.m × stapSMLm × temp/(colspan="2">temp/(colspan="2")temp/(colspan="2")temp/(colspan="2")temp/(colspan="2")temp/(colspan="2")temp/(colspan="2")temp/(colspan="2")temp/(colspan="2")temp/(colspan="2")temp/	73 -	temp = R\(R'\stSteeringVec); % Forward substitution followed by backsubst:	itution	
75 76 77 - y = w'*datavec; 78 - y = y(:); 5 stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m × 5 stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m ×	74 -	<pre>w = temp/(stSteeringVec'*temp);</pre>		
76 %% Calculate output 77 y = w'*datavec; 78 y = y(:); * stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m × stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m ×	75			
77 - y = w'*datavec; 78 - y = y(:); istapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m × stapSMI In x	76	%% Calculate output		
78 - y = y(:); Image: stapSMI.m x stapSMI.m x stapSMI.m x stapSMI.m x Image: stapSMI limit x	77 -	y = w'*datavec;		
stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m × stapSMI Ln 70 Col 1 OVR	78 -	- y = y(:);		
stapSMI.m × stapalgorithmdemo.m × main.m × stapSMI.m × stapSMI In 70 Col 1 OVR				
stapSMI Ln 70 Col 1 OVR	stapSMI.	m × stapaigorithmdemo.m × main.m × stap5MI.m ×		
			stapSMI	Ln 70 Col 1 OVR 🛒
TERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm, Off,	LTERA, AF	RIA. CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & T	m. Off.	

Forward/Backward Substitution

Algorithmic requirements

- m*(k²+k) cmults per QRD
- $200 * (192^2 + 192) = 7.411.200$
- System requirements
 - Update rate 1 ms:
- Parallelism
 - 7.411.200 cmults / (1ms * 250 MHz) => 29.6 cmults in parallel are needed
 - Chosen: 32 parallel cmults = 128 mults = 512 18x18s on SIII or 128 27x27 on SV

Quartus results:

- 82 18x18,
- 14K registers, 11K ALUTs,
- 1.7K MemLUTs
- 5.7 MBit memory
- 246 MHz (using seed sweep)
- => Processing time: 223658 cycles @ 249 MHz = 0.91 ms

© 2010 Altera Corporation—Public



Forward/Backward Substitution



Back Substitution Results

 Top is original data set with jammer at 60 degrees

 Bottom represents the weights after back substitution.
 Note that at 60 degrees you apply filter of -80 dB



© 2010 Altera Corporation—Public


Single Steering Vector STAP Results



DSP Builder Advanced Blockset Simulation Results

© 2010 Altera Corporation—Public

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



STAP Resources

- 1 channel on EP4SGX230 @ 212 MHz
 - ALUTs: 119k (65% of total)
 - Registers: 146k (80% of total)
 - Multipliers: 827 (64% of total)
 - Memory: 8.3 Mbit (57% of total)
 - Latency: ~1 millisecond (QRD + 64 steering vec)
- 4 channels on EP5SGXD8
 - ALUTs: 476k (83% of total)
 - Registers: 584k (52% of total)
 - Multipliers: 3308 (81% of total)
 - Memory:

33 Mbit (60% of total)

© 2010 Altera Corporation—Public

ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS & STRATIX are Reg. U.S. Pat. & Tm. Off. and Altera marks in and outside the U.S.



STAP Radar Design Example Summary

Advanced Blockset simplifies meeting performance

- Design behavioral models and the tool pipelines to meet speed
- Auto generation of HDL based upon timing requirements
- Supports vector processing
- Floating Point Blockset extension
 - Fused datapath allows for floating point in FPGAs
 - Support for Math.h type functions
 - Integer, single and double precision in same model

STAP Design: Matrix Operations

- Steering vector generation
- QR Decomposition
- Single precision complex

© 2010 Altera Corporation—Public



