

# ARC's Customisable Microprocessors



Kami Sehat

Chief Engineer

[kamiar.sehat@arc.com](mailto:kamiar.sehat@arc.com)

- History
- Customisable Microprocessors
- Benefits
- Challenges
- Customisable Multi-Cores
- Summary

<b>1993</b>	<b>Argonaut Software</b> and <b>Nintendo</b> jointly develop a 16-bit processor for 3D games
<b>1994</b>	<b>Argonaut Software</b> creates <b>Argonaut Technologies</b> to focus on 3D graphics hardware, the processing engines had to be customised for each project/customer
<b>1996</b>	<b>Argonaut Technologies</b> develops the <b>Argonaut RISC Core (ARC)</b> extendible and configurable processor in order to lower engineering workload
<b>1997</b>	<b>Argonaut Technologies</b> creates <b>ARC International</b> to further develop and market the ARC processor
<b>2000</b>	<b>ARC International</b> IPO – the <b>ARCtangent A4</b>
<b>2002</b>	The ARC 16/32 ISA and the <b>ARCtangent A5</b>
<b>2003</b>	The <b>ARC 600</b> – 5 stage pipeline
<b>2004</b>	The <b>ARC 700</b> – 7 stage precise pipeline
<b>2005</b>	The <b>ARCmedia</b> – SIMD extensions for audio/video stream processing

- Use HDL pre-processing techniques to enable:
  - **Adding** new components
  - **Taking out** unwanted components
  - **Fine-tuning** components
- Design remains fixed once customisations are done

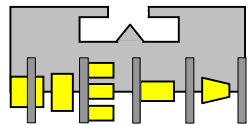
**Customisation = Extension + Configuration**

# Extension: Pre-packaged Components

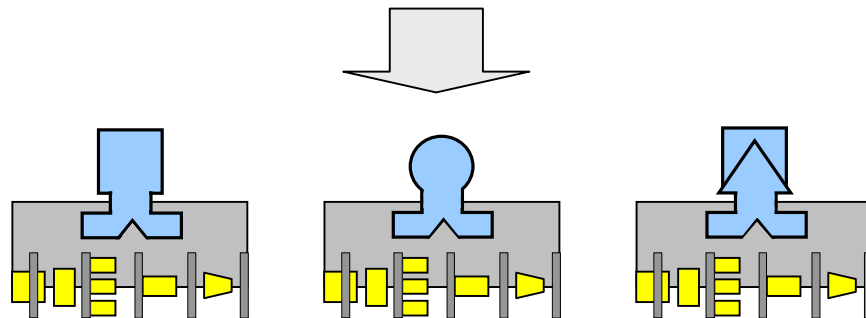
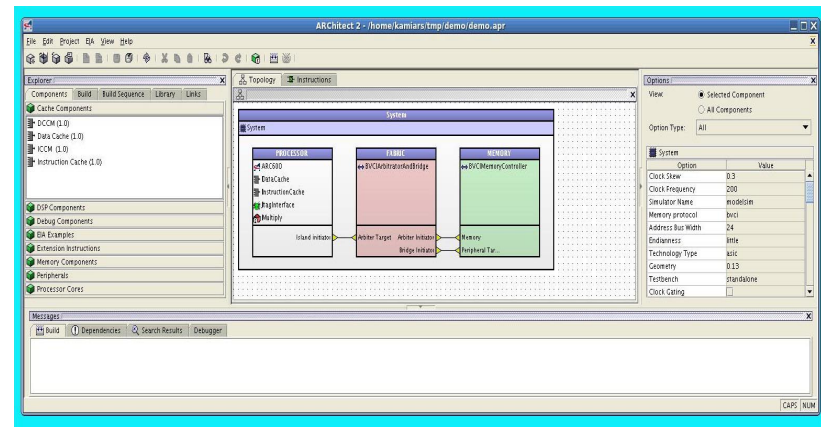
Pre-packaged  
Extension Library



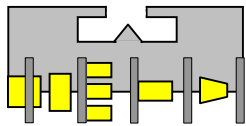
Base case  
Processor



## System Builder



# Pre-packaged Extension Examples



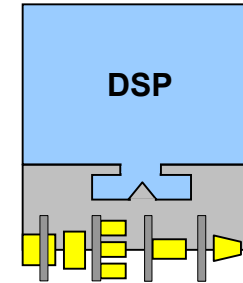
ARC 600

Base case

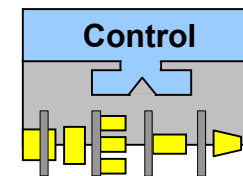
- XY memory +
- SIMD MAC +
- Saturating Round/Shift +
- Absolute +
- Division assist

- ICCM +
- DCCM +
- Extended Interrupts +
- Timers +
- UART

Pre-packaged Extensions



ARC 600 DSP Engine

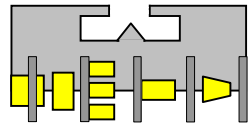


ARC 600 Controller

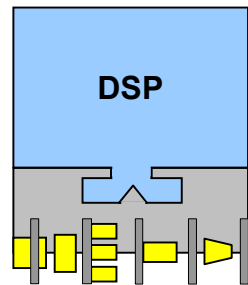
Systems



# Custom-made Extension Examples



ARC 600 Basecase



ARC 600 DSP Engine

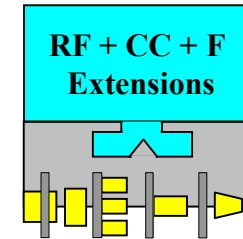
Templates

Additional

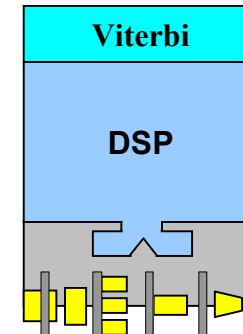
Flags +  
Core Registers +  
Condition Codes

Viterbi Assist  
Instruction

Custom-made  
Extensions



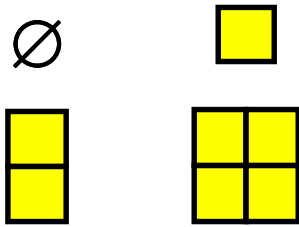
Custom Extended  
ARC 600 Basecase



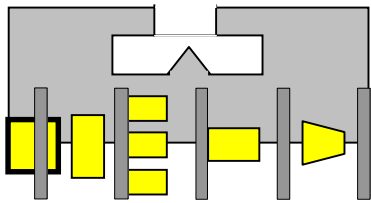
Custom Extended  
ARC 600 DSP

Systems

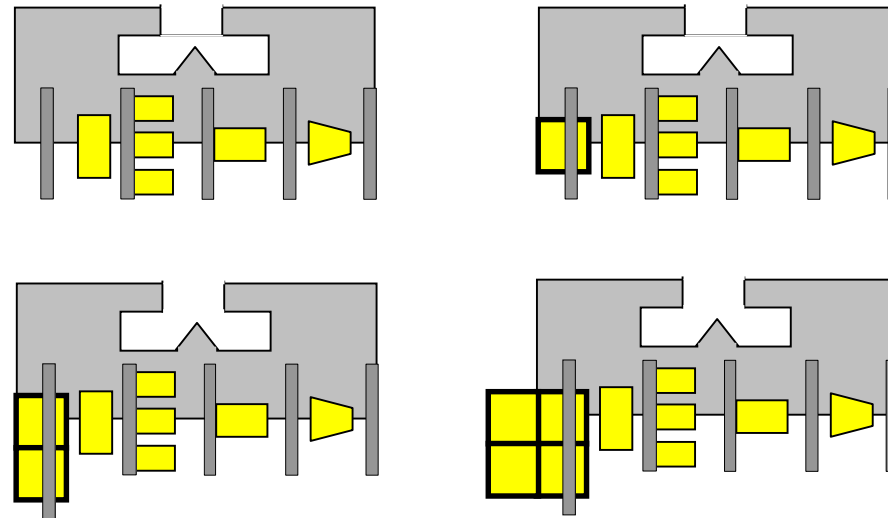
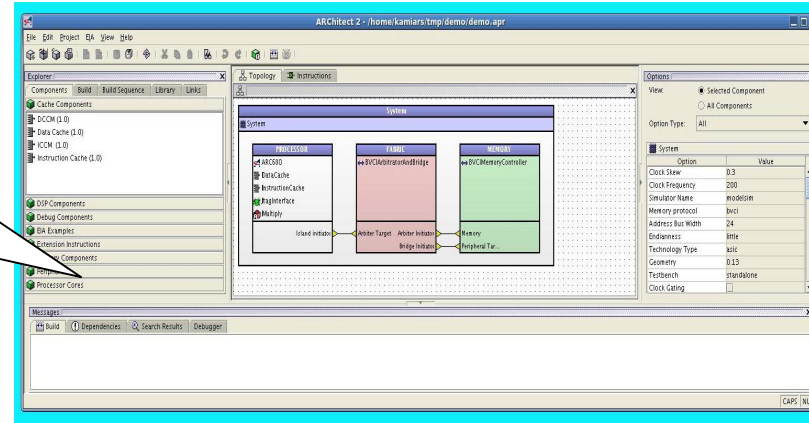
Parameter choices



Base case



## System Builder



# Configuration Examples

Parameter	Range
Instruction Cache Associativity	No Cache, Direct mapped, 2-way or 4-way
Register File Type	32×3-port, 32×4-port, 16×3-port or 16×4-port
Memory Bus Protocol	ARC Proprietary, AHB or BVCI
Implementation Technology	ASIC or FPGA

# Configurable Extension Example

Performance

Small

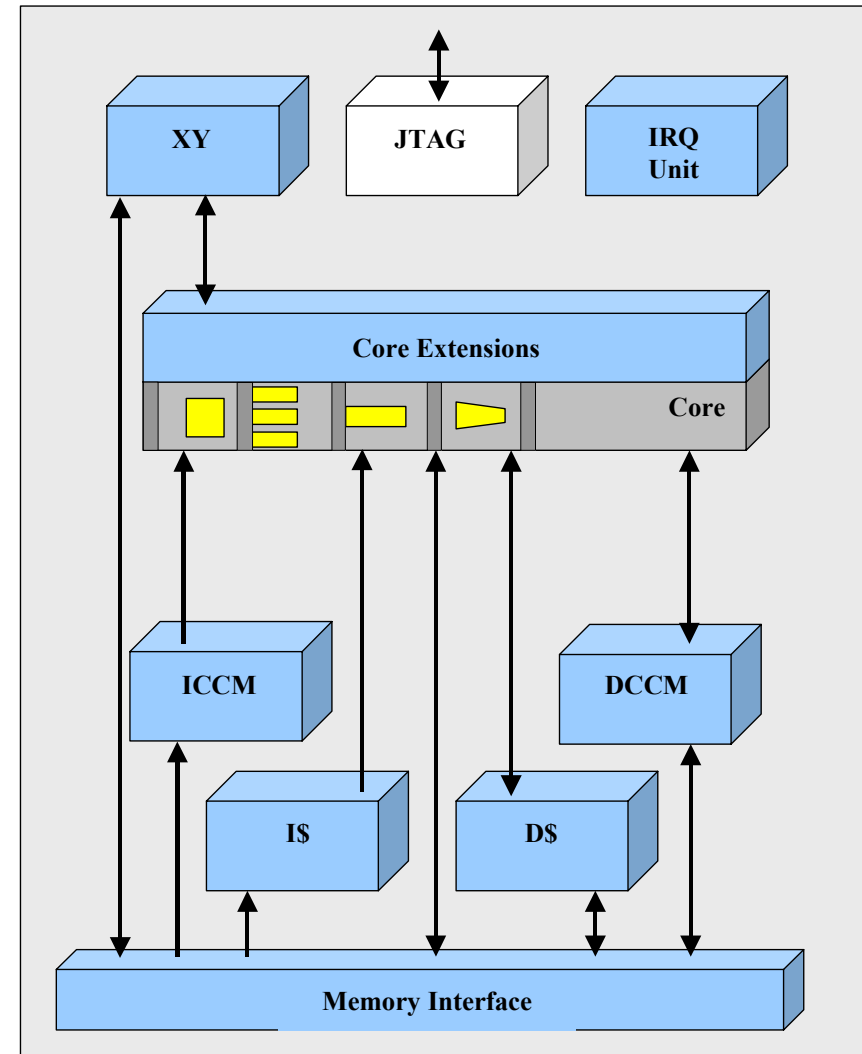
	ARC™ FPX	Floating Point Coprocessor (ARM VF9 S*)
Size in gates	Single Precision: 12K - 14K Double Precision: 24K-30K	100K - 130K
Performance	1.0 Mflops/MHz	1.3 Mflops/MHz
Additional Power	None (confirmed with customer benchmarks)	0.4 mW/MHz typical (0.13um, G process)
Configurability options	Flexible - SP, DP or both	None, monolithic design
Additional math instructions	Can be implemented by the user	Fixed design, not extensible

Flexible

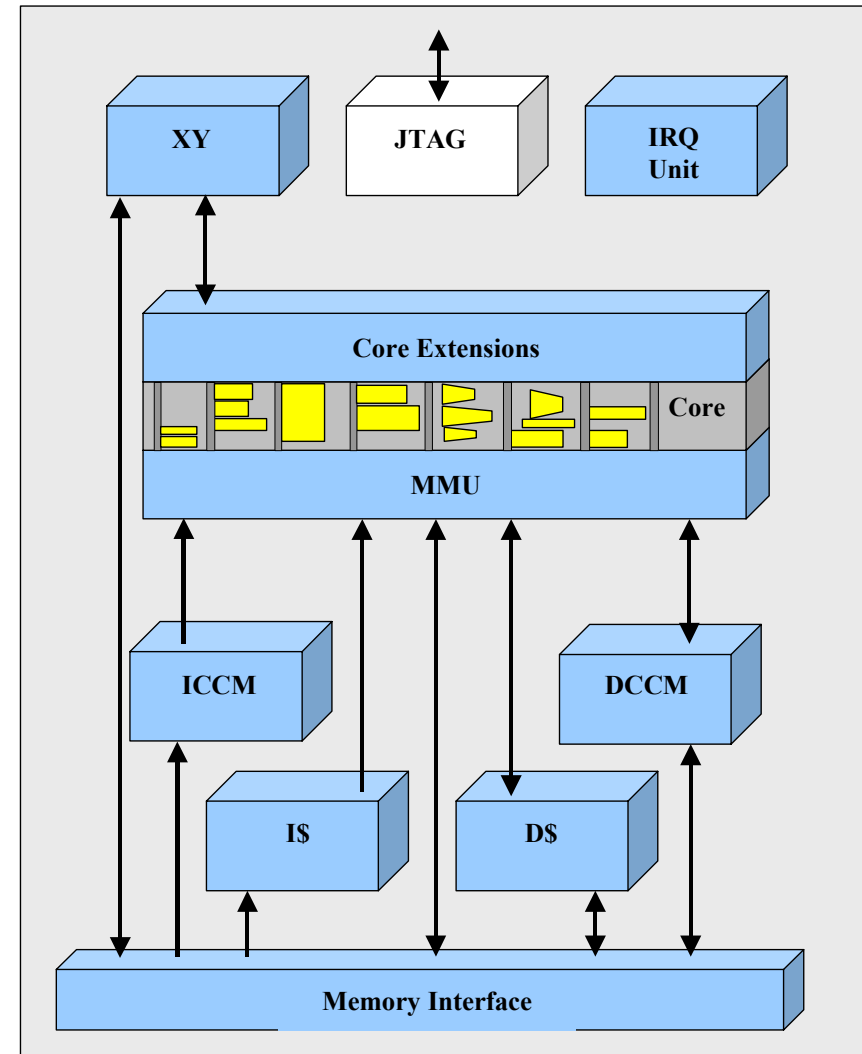
Low Power

\* ARM VFP9-S data from ARM website

Product characteristics in 0.13 $\mu$ m process*	
Clock Frequency	180 - 413 MHz
Silicon Area	0.37 mm <sup>2</sup>
Power Consumption	0.05 mW/MHz
* RAM areas not included.	



Product characteristics in 0.13 $\mu$ m process*	
Clock Frequency	300 - 547 MHz
Silicon Area	0.93 mm <sup>2</sup>
Power Consumption	0.19 mW/MHz
* RAM areas not included.	

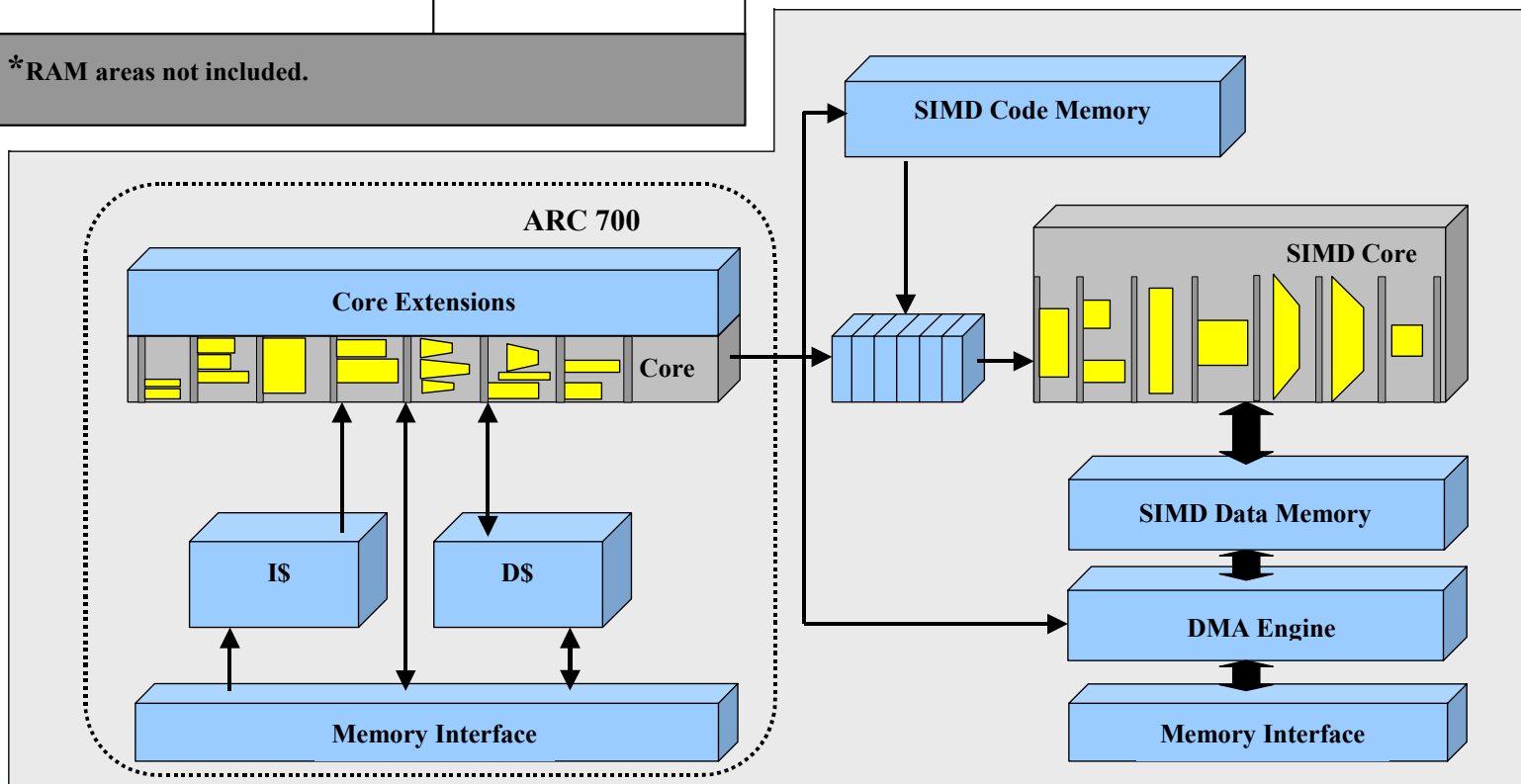


# ARC SIMD Extensions

## Product characteristics in 0.13µm process\*

Clock Frequency	515 MHz
Silicon Area	1.90 mm <sup>2</sup>
Power Consumption	0.40 mW/MHz

\*RAM areas not included.



- Faster Time to market
- Innovations are proprietary to the developer
- Architectural exploration made easy
- Range of cost versus performance options
- Smaller Silicon – i.e. no unused features
- Differentiation of end products
- Single ISA for Multi-Core SoCs

- Customisable Toolset
- Customisable Models
- Customisable Synthesis Flow
- Verification
  - Configuration Combinations
  - Pre-packaged Extensions
  - Custom-made Extensions

In order to be:

- Cost-effective
- Competitive (against fixed IP)

Need to carefully manage customisation

## Customisation Cost Function

---

- Customisable feature cost is more sensitive to productisation overheads than on design complexity
- Grouping of inter-dependent parameters reduces the overheads
- Groupability of a parameter influences its cost

Parameter Selection ↔ Parameter Groupability

- The number of basecase parameters is kept to a minimum
- A large number of optional extension units are provided
- The extension units are highly configurable
  
- Common mistakes when deciding whether to provide flexibility:
  - “The customisable feature is simple to design ...”
  - “The more flexible the better ...”
  - “Let’s leave it open ...”

- A collection of customisable processors, peripherals, arbiters, busses, ...
- Customisable interconnection between components
- An order of magnitude more parameters
- Unacceptable productisation overheads even with parameter selection and grouping
  
- New approaches are required ...

- **Template approach**
  - Pre-configure customisable components
  - Templated components become the Multi-Core building blocks
  - Fine-grain configurability is reduced
  
- **Toolbox approach**
  - Fix and standardise interfaces
  - Provide recipes for putting components together
  - Toolbox contains interface-hardened customisable components and recipes (LEGO style)
  - Systems are not pre-verified

- Customisable processors offer advantages over fixed architectures
- Customisation parameter cost function is more sensitive to productisation overheads than design complexity
- Deciding on the extent and grain of flexibility is the main challenge when designing customisable processors
- New approaches are used to implement customisable Multi-Core and SoCs

