Cambridge Consultants

SAPPHYRE™ DSP

Robert Tan

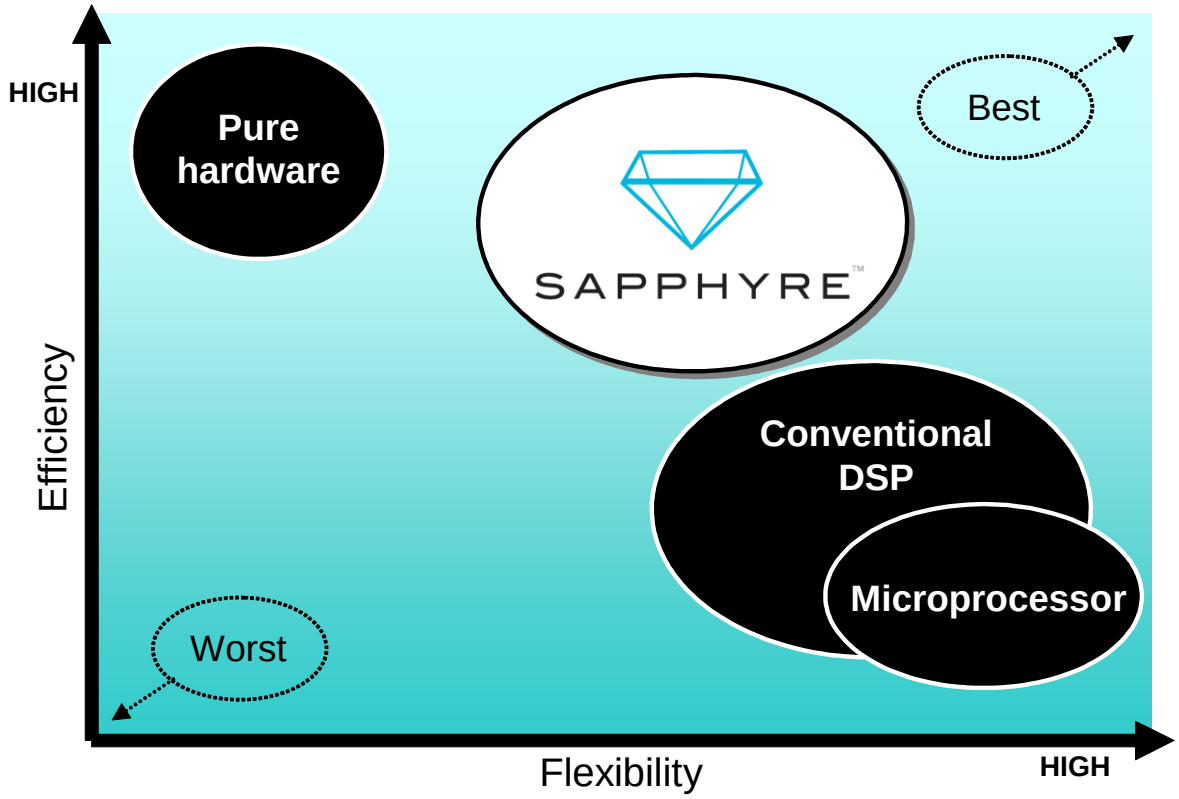
1 March 2017

Sapphyre™ DSP – What is it?

- A suite of technology that Cambridge Consultants use to build application-specific DSP cores for our clients
- The resultant DSP cores can be used in both ASIC and FPGA applications
- The distilled lessons of 100+ signal processing projects
  - Modules and instructions which make the difference for audio, radio, sensing, etc.
- The result of working with customers over years
  - Reducing the costs, timescales and risks of ASIC development
  - Enabling focused and efficient technology, but with firmware upgradability

Sapphyre™ DSP – Why did we develop it?

- Conventional solutions do not support efficient AND flexible processing



Sapphyre™ DSP – Why did we develop it?

- Pure hardware ASIC solutions are efficient but are costly and time consuming to develop, and results are not flexible

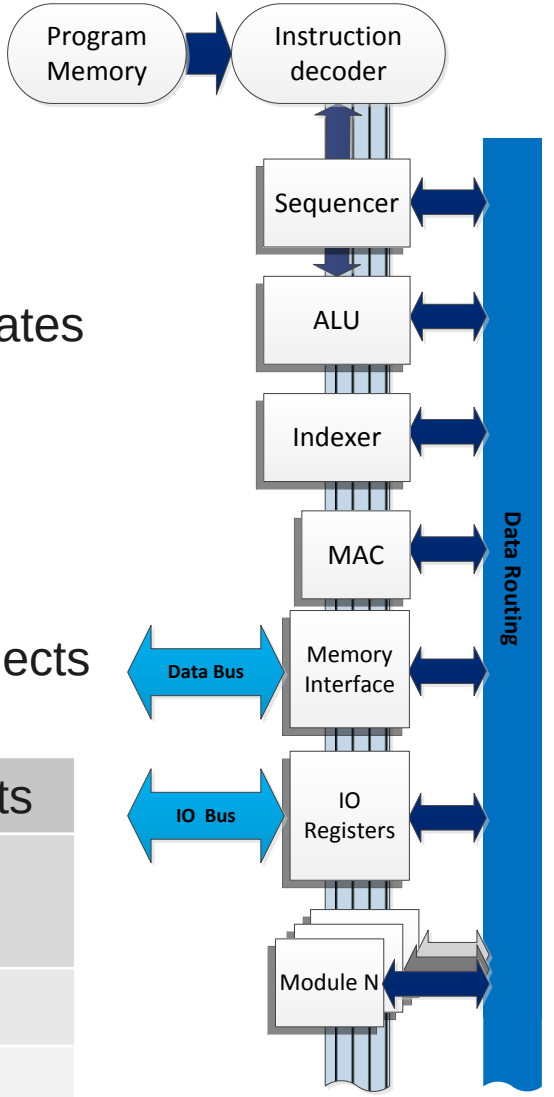
- Conventional DSPs have increased processing capacity by increasing clock rate but this creates unbalanced cores:
  - Processing capacity isn't matched by I/O or memory bandwidth
  - Long pipelines makes it difficult to write efficient processing loops
  - Hardware accelerators are then used to provide efficient operations but are less flexible.

- Sapphyre™ cores specifically run at low clock rates with short pipelines to allow for efficient, flexible DSP code

Sapphyre™ DSP – Why choose it?

- Pure hardware is a good choice if your solution is stable and the benefits of achieving the lowest power point justify the development effort
- Conventional DSPs are a good choice if the mix of processing capacity and hardware acceleration match well to your application
- Sapphyre™ cores are a good choice if:
  - Need to own core
  - Need to save significant BOM cost, relative to conventional DSP
  - Need to save significant power, relative to a conventional DSP
  - Need efficiency close to pure hardware, but need to retain flexibility

Sapphyre™ DSP – What is the structure?



- Each Sapphyre™ DSP core has a VLIW instruction operating many parallel modules each clock cycle
- Low scheduling overheads; efficient processing at low clock rates
- Balanced cores – you can really use the available processing capacity for processing
- Can run autonomously or paired with a host CPU

- Cambridge Consultants has a library of modules allowing projects to get started on creating a custom DSP core quickly

Sequencer	ALU	MAC	Cart2Polar	Constants
Debug monitor	Memory Interface	I/O Registers	Bit operator	Register Bank
Adder	ABS	Sin Cos	Indexer	Min Max
Shifter	Radix	FFT Addr	Oscillator	Limiter

Sapphyre™ DSP – Advantage of simultaneous Core and Code Development

- Cambridge Consultants develop the DSP application code in parallel with the development of the customised Sapphyre™ core

- Developing application DSP code whilst tailoring the core allows quick prototyping of customisations to data routing, module instances, operations or completely new modules
  - Application targeted improvements
  - Balance I/O, memory access and processing
  - Reduces time to develop first product since code can be written before ASIC is complete
  - Reduces ASIC risk as ASIC testing can include confirmation with real application code prior to tape-out

- The resulting Sapphyre™ DSP cores are balanced, efficient designs, tailored to an application but with the flexibility to cope with future expansions

Sapphyre™ DSP – Programmers Toolchain

- Developing code for Sapphyre™ cores is supported by the Programmers Toolchain, consisting of:
  - Macro Assembler
  - Export Tool
  - Graphical Simulator (for Windows)

Sapphyre™ DSP – Macro Assembler

- Allows the Sapphyre™ instructions to be specified in a mnemonic form

```
alu.add ix.q, iy.q;
```

- Supports intuitive short cut notations that further improve readability

```
alu = ix + iy;
```

- Understands the features of each Sapphyre™ design. e.g. What instructions can be combined? How many numeric values can be embedded per program word? etc
- Includes a powerful pre-processor, providing:
  - Single line and block comments
  - Assembler directives such as *.org*, label declarations and numerical symbols
  - C Style directives including; #include, #define, #ifdef, #ifndef, #if, #endif
  - Build time messages and user specified assertion errors
  - Powerful macro (#define) support including; string parameters, numerically evaluated parameters and multi-line macros

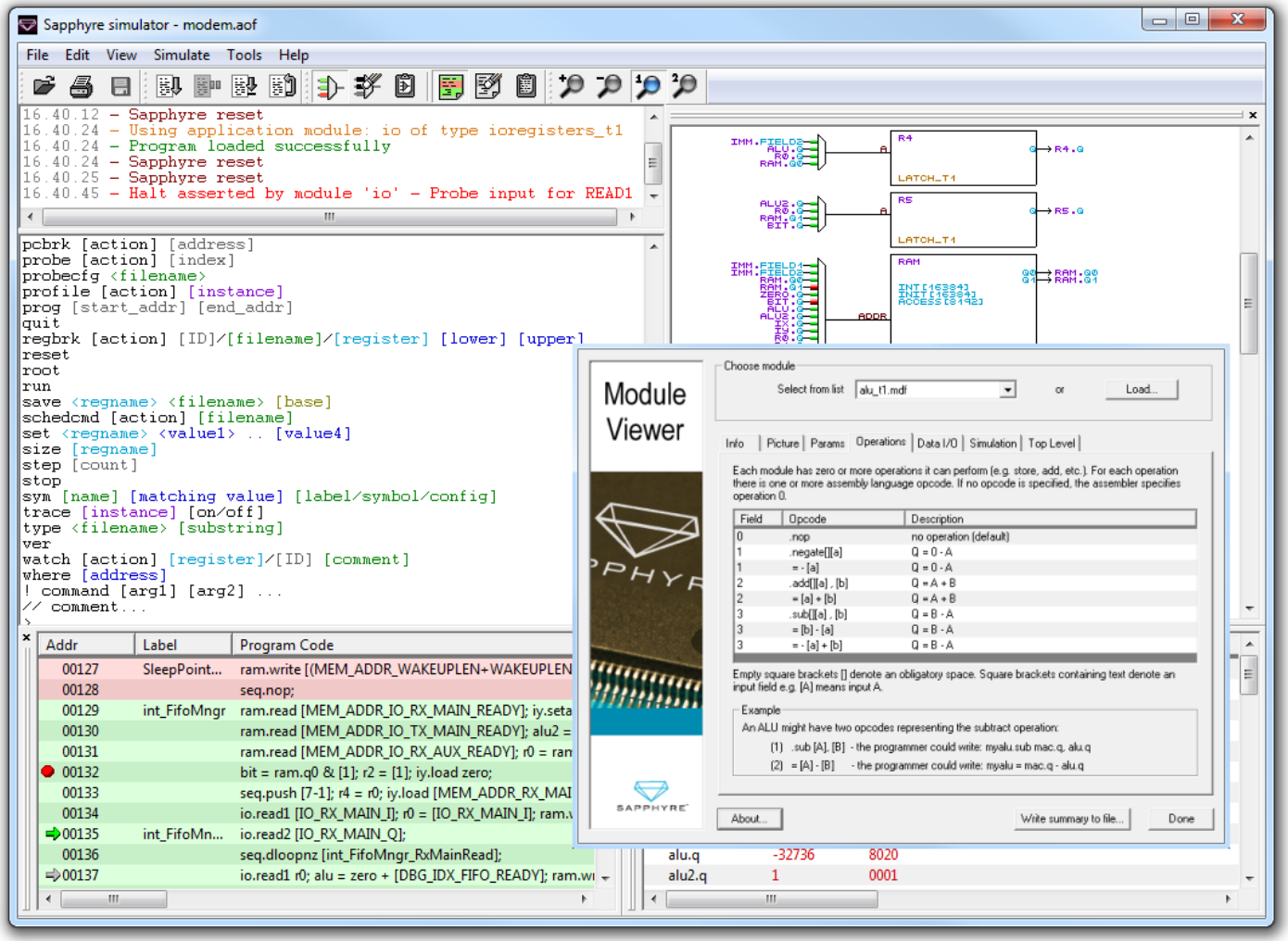
Sapphyre™ DSP – Export Tool

- The Export Tool creates an output file from a template, replacing keyword functions with information from the Sapphyre™ Object File
- Supports creation of interfaces, formatted program ROM and descriptions of the initial RAM state
- Provide formatted access to:
  - The name and value of Sapphyre™ symbols and labels
  - Program memory
  - Initial RAM state
  - Memory sizes
  - Strings parsed from text files

Sapphyre™ DSP – Graphical Simulator

- The graphical simulator is a bit accurate, cycle accurate simulation of a Sapphyre™ core
- Traps on read and/or write access to protected memory regions
- Single stepping, breakpoints, register value breakpoints and register watch windows
- Plugin support for emulation of hardware external to the Sapphyre™ core attached via I/0 registers (e.g. sample interfaces, peripherals, FEC accelerators). *Note: Simple probe file emulation and capture of external I/O is also supported*
- Rich scripting (the simulator can execute a suite of unit tests on a server without any user intervention. This allows automated regression testing with our continuous integration server)
- A variety of profiling tools (code coverage, module usage, VLIW activity, etc.) to indicate the efficiency of the code
- Intuitive visualization of core architecture, used connections, changed values and program flow

Sapphyre™ DSP – Graphical Simulator



Sapphyre™ DSP – The right amount of processing

Sapphyre™ designs can be scaled to provide the optimum balance of computational capacity requirements and power. The range of solutions include:

- Ultra low-power custom cores for sensor applications
- Micro-cores for peripheral interfaces with configurable protocols
- Single MAC cores for audio and stream processing
- Multi-MAC cores for more demanding processing applications (e.g. software defined radio)

Sapphyre™ designs can be used within efficient multi-core clusters to provide even greater processing capacity with highly efficient inter-core communications.

Sapphyre™ works well in chained systems, where a small low-power core operates close to a sensor or interface and only enables the more powerful processing components when specific conditions are detected.

Sapphyre™ DSP – Real-time debug monitoring output

- Not solved well on other systems
- Single cycle stepping doesn't help with real time I/O

A better approach:

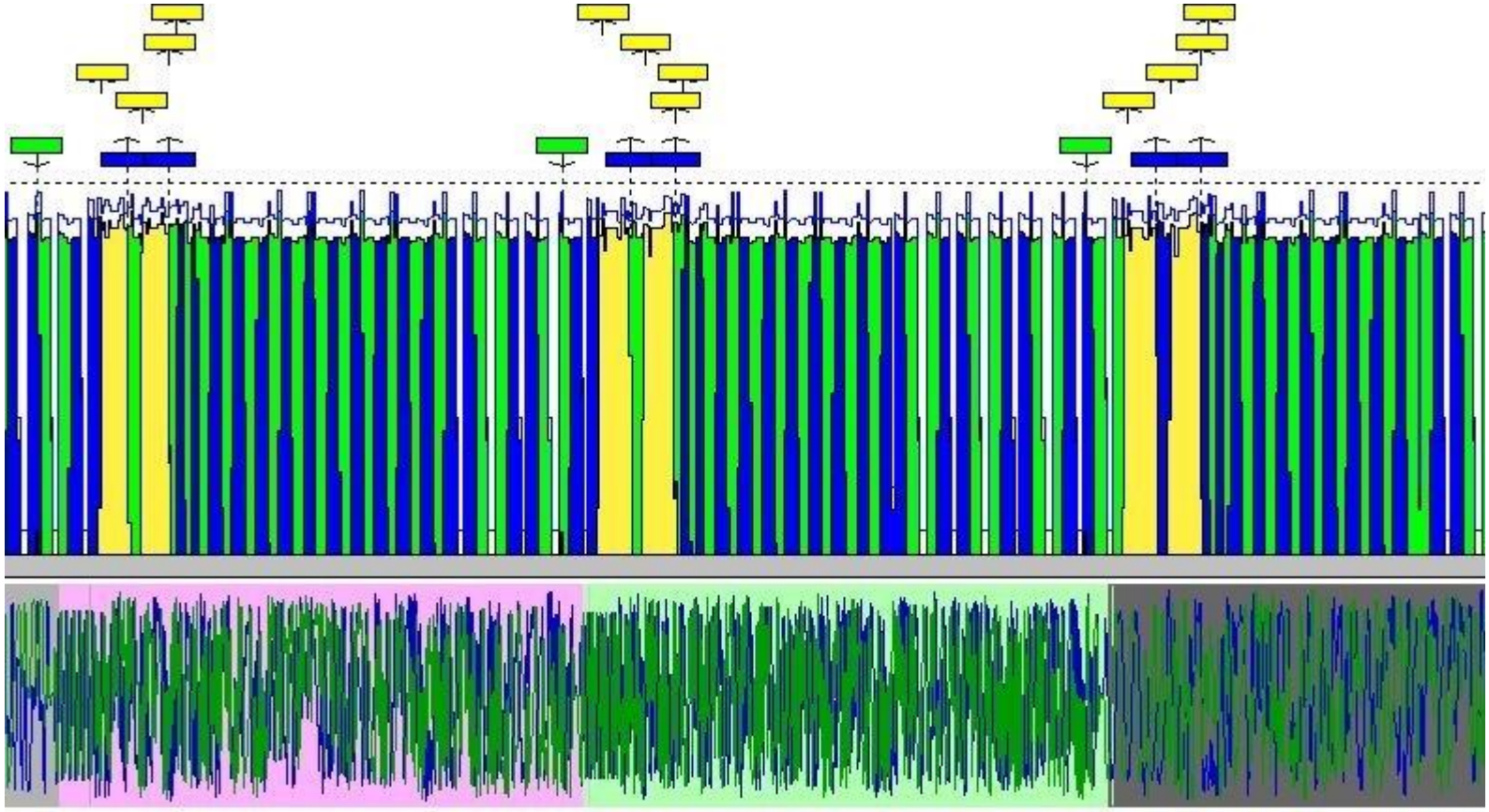
- Test point monitoring of inputs, configuration and intermediate outputs
- Replay and debug in simulator with access to many more internal states
- Operates well with multi-core DSP debugging
- Parallel and non-invasive (doesn't change DSP operation)
- Can be used to measure processor loading, latency and performance
- Helps to optimise and make full use of processing capacity

Sapphyre™ DSP – Real-time debug monitoring output

Events

Tasks / Load

Input Signal



Sapphyre™ DSP – FPGA prototyping, emulation and products

- The Generator templates for Sapphyre™ work well in both ASIC and FPGA
- Whilst the FPGA Sapphyre™ implementations consume more static power than ASICs and have high unit costs they may be suitable for some applications
- Sapphyre™ cores have a small footprint and use of low clock rates which means that modern FPGA's can achieve processing speeds similar to ASICs
  - Fast development of FPGA prototype using Cambridge Consultant's library of Sapphyre™ modules
  - Sapphyre™ designs for ASIC can use FPGA's to create real-time, or near real-time ASIC emulation
    - Emulation of whole cores/clusters
    - Significantly reduces risk of re-spin
- Can use off-the-shelf FPGA PCI cards

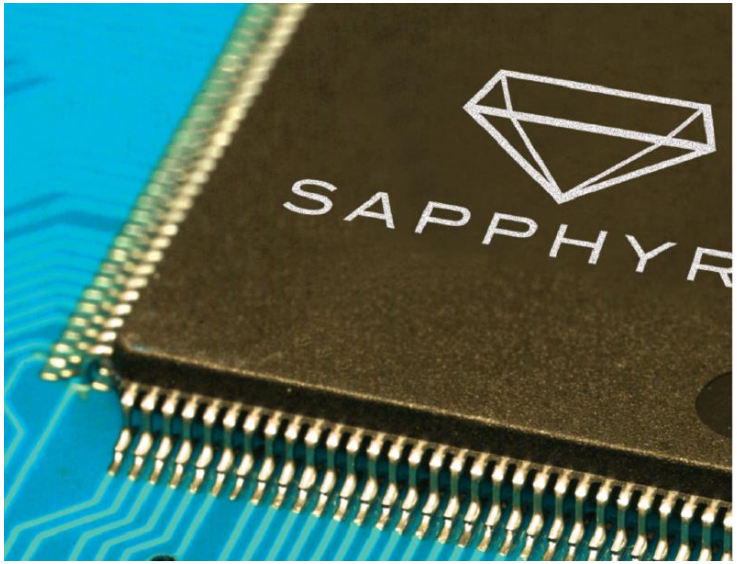
Sapphyre™ DSP – What applications is it good for?

- High performance/real time audio processing e.g. codecs
- Software defined radio
- Ultra-low power processing
- Taking cost out of projects: Can replace $10s of DSP with cents of silicon. With low risk, low cost and fast development, ASICs can pay back even at low volumes
- Offloads maths-intensive functions from a CPU
- Offers a “FPGA now, ASIC later” roadmap
- Offers a “soft now, hardened later” roadmap
  - Program RAM to metal layer ROM, then to crunched ROM
- Development of ASIC IP blocks (co-processors, functional accelerators, codec accelerators, re-configurable interfaces)
- Where clients want to own the processing core

Sapphyre™ DSP – Want to know more?

- If you are interested in using Sapphyre™ to provide a low power DSP solution for your next project please contact us at:

sapphyre@cambridgeconsultants.com



Cambridge Consultants