



The BondMachine, a mouldable computer architecture

Mirko Mariotti¹, Daniel Magalotti²

Introduction

The BondMachine (BM) is a new computer architecture where many Connecting Processors (CPs) with different Instruction Set Architecture(ISA) are connected together and share resources to form an **heterogeneous** ensemble perfectly fitted to a specific computational problem. These cores are implemented with the characteristic to be as minimal as possible and as simple as possible, and the capacity of solving problems rely mainly in how they are **interconnected**.

A BondMachine architecture can also grown by using evolutionary algorithms that select the architectures, processors programs and interconnections. In order to satisfy and improve the power processing, the BondMachine is implemented by using the Field Programmable Gate Array (**FPGA**) chips, that are today's most powerful implementations of reconfigurable hardware. Moreover the "register machine" abstraction has been kept in order to use many well known tools and techniques ranging from languages to compilers.

This architecture can be used as general purpose computer architecture or as high specialized device perfectly suited to specific problems and flexible enough to be used in scenarios like Internet of Things (**IoT**)[1], Cyber Physical System (**CPS**)[2] and High Performance Computing (**HPC**).

The BondMachine architecture

The BondMachine architecture consists of interconnections among Connecting Processors and Shared Objects (SOs) build to implement a dedicated tasks. The main features of this kind of architecture are the possibility to configure:

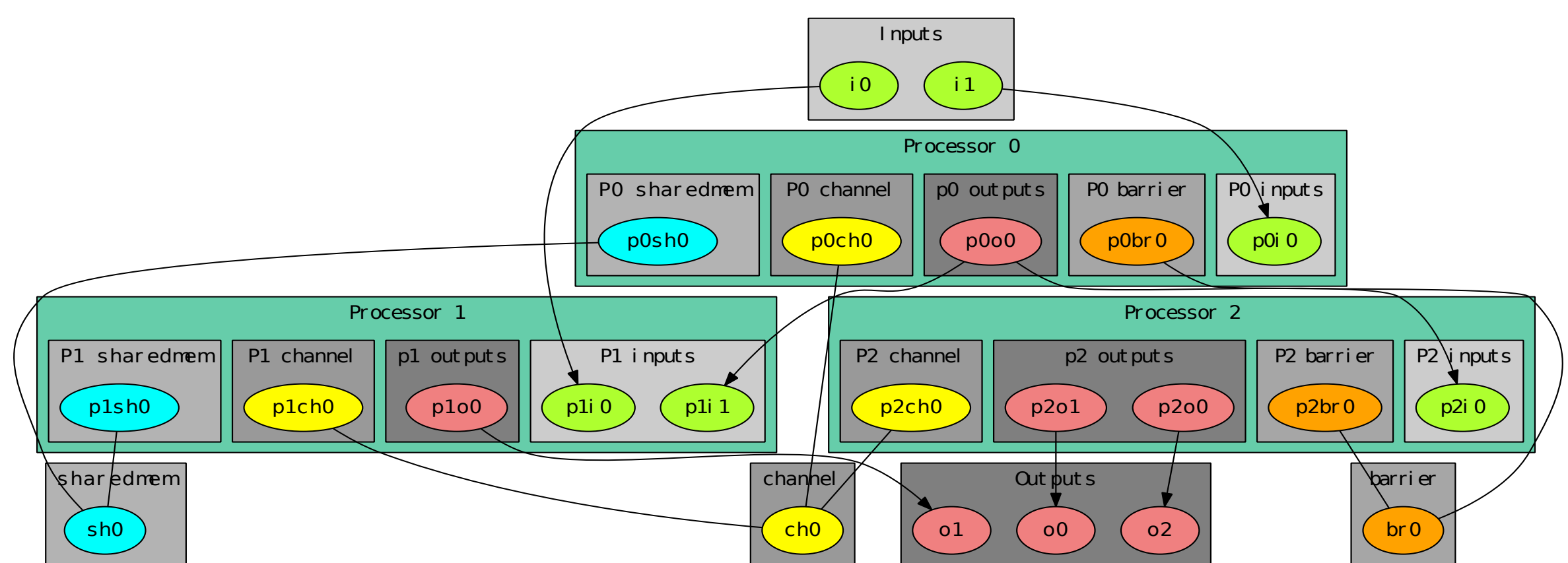
- the number of processor cores and their types,
- the number of inputs and outputs,
- the interconnection between processors,
- the number and the type of SOs used by each processor.

Connecting Processor

The CP is the **computing core** of the BondMachine. Several CPs can be configured in arbitrary connection topologies within the BondMachine. They can have different registers number, istruction set, io-registers with respect to the other ones.

Shared Object

Any kind of component the can be **shared** among CPs. Shared Objects increase the processing capability and the functionality of the BM improving the high-speed **synchronization** and **communication** between tasks running on separate CPs.



A complete example of the BondMachine architecture. It consists of two inputs and tree outputs interconnected between the input/output registers of the processors. Shared objects, such as memory, Channel and Barrier, are connected among the processors.

Software Tools

The complexity of programming the BondMachine architecture is managed by using a set of software tools to:

- build a specify architecture as function of the task,
- modify the created architecture,
- simulate the behavior and to check the functionality with the aim to generate the Register Transfer Level (RTL) code for FPGA device.

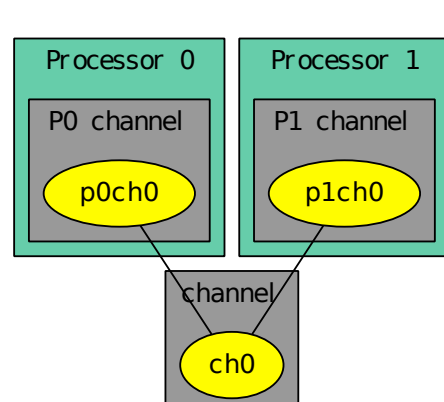
Processor Builder selects the CP specifics, assembles and disassembles, saves on disk as JSON, emulates and creates the RTL code.

BondMachine Builder connects CPs and SOs together in custom topologies, loads and saves on disk as JSON, emulates and creates the RTL code.

Arch-compiler compiles the Go language to generate the CP assembly code and to create the optimized architecture to run that code.

Case study

This example is a simple scenario with two CPs that send a data back and forth through a Channel. The Processor0 sends the data through the Channel, the Processor1 receives it and sends it back by using the same Channel. When the Go source code is compiled the Bondgo Arch-compiler produces the **architecture** specific to the problem, **optimized** (only the needed opcodes are produced, different ISA) for it and **the assembly code** to run on it.



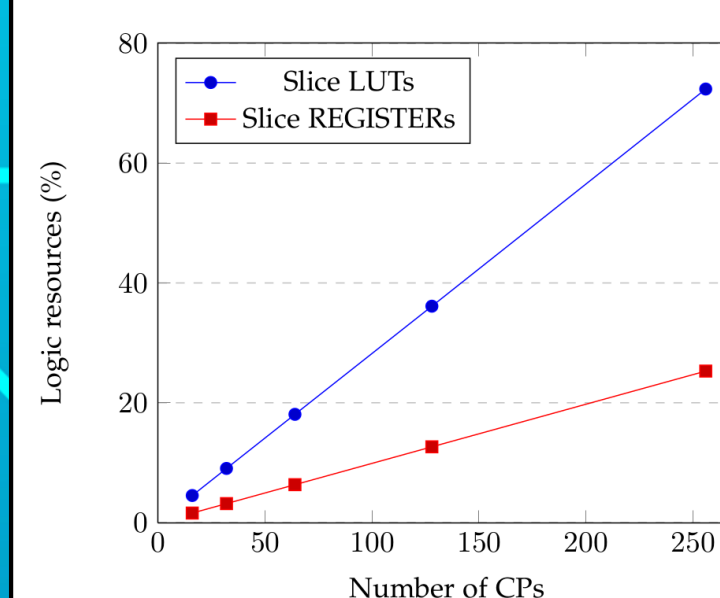
Processor 0
000000000000 c1r r0
001000000000 r2m r0 0
010000000001 rset r0 1
001000000000 r2m r0 0
011000000000 m2r r0 0
100000000000 wwr r0 ch0
101100000000 chw r1
110000000000 wrd r0 ch0
101100000000 chw r1
001000000000 r2m r0 0
001000000000 r2m r0 0
111010000000 j 4

Processor 1
00000000 c1r r0
00100000 r2m r0 0
01000000 wrd r0 ch0
01110000 chw r1
00100000 r2m r0 0
10000000 m2r r0 0
10100000 inc r0
00100000 i2m r0 0
10000000 m2r r0 0
11000000 wwr r0 ch0
01110000 chw r1
1110010 j 2

```
package main
import (
    "bondgo"
)
func pong(c chan uint8) {
    var ball uint8
    for {
        ball = <-c
        ball++
        c <- ball
    }
}
func main() {
    ball:=uint8(1)
    ch := make(chan uint8)
    go pong(ch)
    for {
        ch <- ball
        ball = <-ch
    }
}
```

Hardware implementation

The RTL code automatically generated by the builders is synthesized for the FPGA XC7K325T (Xilinx KC705 evaluation card) to measure the **performances** of the architecture: logic resources, power consumption and maximum clock frequency rate.

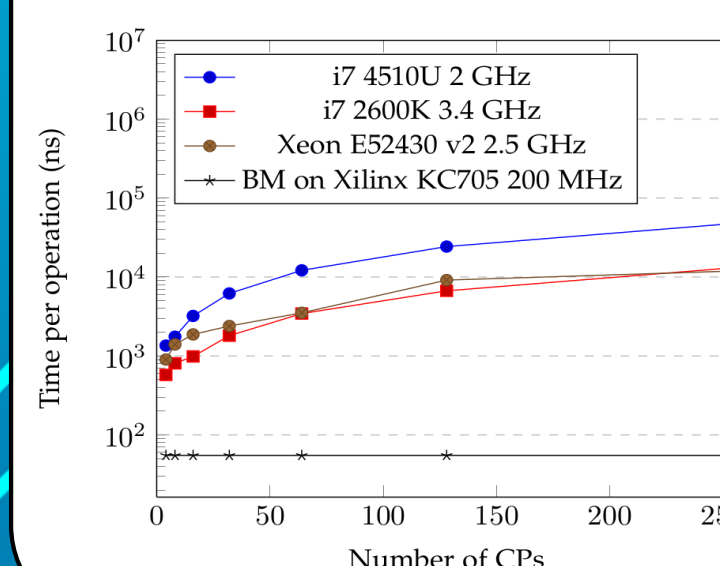


The architecture consists of a channel shared by two CPs.

This basic element has been replicated by varying the number of CPs and Channels:

- the logic resources used by each architecture increase linearly as function of CP.

The FPGA can contain up to 256 CPs with a clock frequency of 200 MHz and a power consumption of 6.13 W.



The performances of the architecture have been compared with the Go ones. A benchmark has been used to measured the time per operation needed to the architecture to complete the task.

The different performances of the architecture show:

- the time per operation increases linearly for the CPU, due to the fact that the number of emulated CPs that works in parallel increases;
- the time per operation is constant for the FPGA due to the **intrinsic parallelism** (up to fill all the available logic resources).

Evolutionary BondMachine

Some particular problem may need a complex network of CPs and Shared Objects to be solved especially regarding the internal interconnections and the feature to have processor of different types.

The BondMachine emulator has been connected to MEL (My Evolutionary Language), an Evolutionary Computing Framework to explore the possibility of **evolving the architectures** to solve a specific problem.

Conclusion

The BondMachine is a new kind of computing device made possible in practice only by the emerging of new re-programmable hardware technologies such as FPGA. Keeping the register machine abstraction it is possible to borrow well known languages and techniques in programming these devices removing the need of having a general purpose architecture. Moreover the BondMachine architecture is high specialized device perfectly suited to specific problems and flexible enough to be used in many scenarios finding the better topology of interconnection's processors.