A systems re-engineering case study
Programming robots with occam and Handel-C

Dan Slipper

Supervisors:
Alistair A. McEwan
Wilson Ifill

AWE
The Problem
Re-engineering

Platform Independent Model

Legacy System Component

Replacement Component

Behaviour

Architecture
LEGO Mindstorms NXT

- 32-bit ARM7 microprocessor
- 4 input ports
- 3 output ports
- 100x64 pixel LCD
- Bluetooth
- Power, 6 AA Batteries

Re-engineering
Case Study

CSP Program Structure

LEGO MINDSTORMS NXT

Replacement Hardware System

```
PAR
  motorASpeed ! 70
  motorBSpeed ! 70
reqLight ! 0
fromLight ? light
WHILE light > 800
  SEQ
    reqLight ! 0
    fromLight ? light
  PAR
    motorASpeed ! 0
    motorBSpeed ! 0
do.delay(2000)
```
Case Study

CSP Program Structure

occam

NXT

Handel-C

FPGA
Overview

• Problem space
• Project aims and background
• NXT/Transterpreter implementation
• FPGA implementation
• Case study
• Conclusions and future work
Aims

• Investigate the impact of modelling at a platform independent level

• Introduce two platforms using a common model of concurrency
  – Running a simple common task

• Demonstrate behavioural differences and integration issues between implementations
FPGAs and Handel-C

• FPGAs are “reconfigurable hardware”
• Can be reprogrammed any number of times unlike ASIC
• Handel-C is a HDL making FPGA hardware programming look like software
• C like language augmented with a CSP model of concurrency
Handel-C Example

static macro proc P();
static macro proc C();
void main (void)
{
    chan int 1 chanA;
    chan int 1 chanB;
    par{
        P(chanA, chanB);
        C(chanB, chanA);
    }
}
occam and the Transterpreter

• The Transterpreter is a modern virtual machine for a variety of commodity platforms
• Interprets bytecode as on the Transputer
• Written as a C library
PROC main ()

CHAN INT chanA, chanB:

PAR
  P(CHAN INT chanA, chanB)
  C(CHAN INT chanB, chanA)
:

occam Example
Case Study

CSP Program Structure

Legacy System Component

x

Replacement Component

occam

Transterpreter

NXT

Handel-C

FPGA
NXOS + Transterpreter

• Built on top of existing OS
  – NXOS – set of C drivers and boot code
Peripheral Support in occam

• Standard peripherals have support
  – Analogue sensors
  – Motors
  – Ultrasonic sensor
  – Speaker
  – LCD
  – Bluetooth

• NXT 2.0
  – Colour sensor currently unsupported
  – 3rd party sensors partially supported – Using I²C
FPGA Architecture

- Ultrasound Sensor
- Light Sensor
- Touch Sensor
- Sound Sensor
- Motors (with tachometer)
FPGA features

• Light/Sound/Touch
  – Implemented through ADC

• Motors
  – PWM
  – Tachometer
  – H-Bridge driver board
Case study

• Use implementations on FPGA and NXT
• Common program architecture
• Utilising a number of peripherals
• Simple case study
  – Follow a set path around the desk, changing direction with different speeds, (therefore angles)
  – Recognise the edge of the desk
Path to follow

• The path the robots should follow...
Design of Experiment
Robot Structure
Process / Communication Structure
Main Implementation

• occam

PROC main ()

CHAN INT motorASpeed, fromLight, reqLight, fromTachA, reqTachA, reqTachB:

...

PAR

LightSensor(fromLight, reqLight)
Motor1(motorASpeed)
Motor1Tach(fromTachA, reqTachA)
ControlCode(motorASpeed, fromLight, reqLight, fromTachA, reqTachA)

:

• Handel-C

void main (void)
{
    chan int 32 toMotorA;
    chan int 32 motorATach;
    chan int 1 reqTachA;
    par{
        ADC_Read();
        LightSensor(fromLight, reqLight);
        Motor1(toMotorA);
        Motor1Tach(motorATach, reqTachA);
        ControlCode(fromLight, toMotorA, reqLight, reqTachA, motorATach);
    }
    }
}
Control Implementation

**occam**

PAR
  motorASpeed ! 70
  motorBSpeed ! 70
reqLight ! 0
fromLight ? light
WHILE light > 800
  SEQ
    reqLight ! 0
    fromLight ? light
PAR
  motorASpeed ! 0
  motorBSpeed ! 0

**Handel-C**

par{
    toMotorA ! 70;
    toMotorB ! 70;
}
reqLight ! 0;
fromLight ? light;
while(light == 0)
{
    reqLight ! 0;
    fromLight ? light;
}
par{
    toMotorA ! 0;
    toMotorB ! 0;
}
Test Results

• Path from occam implementation was a reference
• Handel-C path was wrong
  – Turning angles were different
  – Tachometer readings were therefore different
  – Overall system behaviour was incorrect
Further Tests

- Pulses after travelling 1000
  - Speed 100%

<table>
<thead>
<tr>
<th>Robot</th>
<th>Run 1</th>
<th>Run 2</th>
<th>Run 3</th>
<th>Run 4</th>
<th>Run 5</th>
<th>Mean</th>
<th>Range</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handel-C</td>
<td>1247</td>
<td>1240</td>
<td>1239</td>
<td>1233</td>
<td>1232</td>
<td>1238</td>
<td>15</td>
<td>6.058052</td>
</tr>
<tr>
<td>occam</td>
<td>1773</td>
<td>1783</td>
<td>1793</td>
<td>1814</td>
<td>1787</td>
<td>1790</td>
<td>41</td>
<td>15.26434</td>
</tr>
</tbody>
</table>

- Speed 80%

<table>
<thead>
<tr>
<th>Robot</th>
<th>Run 1</th>
<th>Run 2</th>
<th>Run 3</th>
<th>Run 4</th>
<th>Run 5</th>
<th>Mean</th>
<th>Range</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handel-C</td>
<td>1213</td>
<td>1208</td>
<td>1209</td>
<td>1203</td>
<td>1206</td>
<td>1208</td>
<td>10</td>
<td>3.701351</td>
</tr>
<tr>
<td>occam</td>
<td>1397</td>
<td>1417</td>
<td>1431</td>
<td>1415</td>
<td>1413</td>
<td>1415</td>
<td>34</td>
<td>12.1161</td>
</tr>
</tbody>
</table>
Further Tests (2)

- Distance travelled (cm)
  - Speed 100%

<table>
<thead>
<tr>
<th>Robot</th>
<th>Run 1</th>
<th>Run 2</th>
<th>Run 3</th>
<th>Run 4</th>
<th>Run 5</th>
<th>Mean</th>
<th>Range</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handel-C</td>
<td>73.8</td>
<td>71.3</td>
<td>68.6</td>
<td>69.6</td>
<td>68.6</td>
<td>70.38</td>
<td>5.2</td>
<td>2.207261</td>
</tr>
<tr>
<td>occam</td>
<td>70.4</td>
<td>70.4</td>
<td>71.0</td>
<td>70.6</td>
<td>71.2</td>
<td>70.72</td>
<td>0.80</td>
<td>0.363318</td>
</tr>
</tbody>
</table>

- Distances average the same over a vast range
- Investigation is required into circuit and differences in hardware
Further Tests (3)

Average motor Voltage against Time when turning 1000 degrees

Voltage (V) vs. Time (Seconds)

- RED: NXT Voltage
- BLUE: FPGA Voltage
Results

• Voltage difference between hardware implementations
  – Larger H-Bridge circuit
• Braking only tested with ‘float’ method
  – void nx_motors_stop(U8 motor, bool brake)
Conclusions and Future Work

Conclusions
• Experiments demonstrate that programming same high-level missions leads to different behaviours.
• Therefore just modelling the high level behaviours is not reliable enough for a systems re-engineering project
• Modelling and verification methods are required for the whole system

Future Work
• CSP model of system behaviour
  – Translation to both implementation languages
• Improve motor drivers to braking methods
• More complex case study