LEGO RCX
Hitachi H8/3292

LISHA/UFSC
Fauze Valério Polpeta
Prof. Dr. Antônio Augusto Fröhlich
{fauze|guto}@lisha.ufsc.br
http://www.lisha.ufsc.br/~{fauze|guto}

March 2003
LEGO RCX Overview

- LEGO RCX
  - Programmable hardware module
  - Interface to I/O devices (sensors and actuators)
  - Hitachi H8/3292 microcontroller
LEGO RCX I/O Devices

- **Input devices**
  - 3 sensor ports (through A/D converters)
  - 4 buttons
  - battery level monitor
  - timers

- **Output devices**
  - 3 actuator ports (through D/A converters)
  - 5-segment LCD
  - speaker

- **Bidirectional devices**
  - infrared port
Hitachi H8/300 Overview

- **H8/300 CPU**
  - 8-bit data
  - 16-bit address space
  - 8 x 16 bit GP registers (r0 - r7)
    - r0 => function return
    - r7 => stack pointer
  - 16 MHz clock
Hitachi H8/3292 Block Diagram

- **CPU**
  - H8/300

- **H8/3292**
  - **ROM**
    - 16 KBytes
    - (0x0000 – 0x3fff)
  - **on-chip RAM**
    - 512 Bytes
    - (0xfd80 – 0xff7f)
  - **on-chip Register Field**
    - 120 Bytes
    - (0xff88 – 0xffff)

- **RAM**
  - 28 Kbytes
  - (0x8000 – 0xefff)

- **off-chip Register Field**
  - ? bytes
  - (0xf000-0xf????)
Hitachi H8/300 Address Modes

- Address modes
  - Register direct: rn
  - Register indirect: @rn
  - Register indirect with 16-bit displacement: @(d:16,rn)
  - Register indirect with post-increment: @rn+
  - Register indirect with pre-increment: @-rn
  - Absolute address (8 or 16 bits): @aa:8, @aa:16
  - Immediate (8 or 16-bit data): #aa:8, #aa:16
  - PC-Relative (8-bit displacement): @(d:pc)
  - Memory indirect: @@aa:8
LEGO RCX Memory Layout

- **MCU mode 2 (control register at 0xffffc5)**

<table>
<thead>
<tr>
<th>Address Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0000</td>
<td>on-chip ROM</td>
</tr>
<tr>
<td>0x3fff</td>
<td>H8/3292 interrupt vector table, RCX firmware</td>
</tr>
<tr>
<td>0x8000</td>
<td>off-chip RAM</td>
</tr>
<tr>
<td>0xefff</td>
<td>Application program/data</td>
</tr>
<tr>
<td>0xff7f</td>
<td>on-chip RAM</td>
</tr>
<tr>
<td>0xffff</td>
<td>Register Field</td>
</tr>
<tr>
<td>0x8000</td>
<td>on-chip RAM</td>
</tr>
<tr>
<td>0xefff</td>
<td>RCX devices</td>
</tr>
<tr>
<td>0xf7f</td>
<td>RCX interrupt vector table</td>
</tr>
<tr>
<td>0xffffffff</td>
<td>H8/3292 devices</td>
</tr>
</tbody>
</table>
LEGO RCX Interrupt Dispatching

- **H8/3292 interrupt table**
  - Stored at \(0x0000-0x0049\) (ROM in the RCX)
  - RCX ROM vectors redirect interrupts to the on-chip RAM interrupt table
  - Decreasing priority

- **on-chip RAM interrupt table**
  - Stored at \(0xfd80-0xfdbf\)
  - Pointers to user-defined handlers

- **Masking**
  - Globally (except NMI) CCR bit 7
  - Individually through the off-chip register field

<table>
<thead>
<tr>
<th>Vector</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>reset</td>
</tr>
<tr>
<td>1 - 2</td>
<td>reserved</td>
</tr>
<tr>
<td>3</td>
<td>NMI</td>
</tr>
<tr>
<td>4 - 6</td>
<td>IRQs</td>
</tr>
<tr>
<td>7 - 11</td>
<td>reserved</td>
</tr>
<tr>
<td>12 - 18</td>
<td>16-bit timer</td>
</tr>
<tr>
<td>19 - 21</td>
<td>8-bit timer 0</td>
</tr>
<tr>
<td>22 - 24</td>
<td>8-bit timer 1</td>
</tr>
<tr>
<td>25 - 26</td>
<td>reserved</td>
</tr>
<tr>
<td>27 - 30</td>
<td>serial</td>
</tr>
<tr>
<td>31 - 34</td>
<td>reserved</td>
</tr>
<tr>
<td>35</td>
<td>ADI</td>
</tr>
<tr>
<td>36</td>
<td>WOVF</td>
</tr>
</tbody>
</table>
Interrupt Dispatching

Diagram showing a system with:
- CPU: H8/300
- ROM
- RAM
- on-chip RAM
- on-chip Register Field
- off-chip Register Field
- running code

Software/Hardware Integration

March 2003 http://www.lisha.ufsc.br
Interrupt Dispatching

- H8/3292
- ROM
- dispatcher
- CPU H8/300
- on-chip RAM
- on-chip Register Field
- RAM
- interrupted code
- off-chip Register Field
Interrupt Dispatching

CPU H8/300

H8/3292

ROM

dispatcher

RAM

pointe

on-chip RAM

on-chip Register Field

interrupted code

off-chip Register Field

handler
Interrupt Dispatching

- H8/3292
- ROM
- CPU H8/300
- on-chip RAM
- on-chip Register Field
- off-chip Register Field
- RAM
- handler
- dispatcher
- interrupted code
Interrupt Dispatching

- CPU H8/300
- H8/3292
- ROM
- dispatcher
- on-chip RAM
- on-chip Register Field
- RAM
- running code
- off-chip Register Field
LEGOSTX Interrupt Handling

- **H8 dispatching**
  ```c
  push pc
  push ccr
  ccr[7]=1 /* int disable */
  ```

- **H8/300 Handler (ROM)**
  ```c
  void h8_handler(void) {
    push r6
    mov  RCX_Int_Table[n],
    r6
    jsr  @r6
    pop  r6
    rte
  }
  ```

- **RCX Handler**
  ```c
  void rcx_handler(void) {
    /* push registers */
    /* handle interrupt */
    /* pop registers */
  }
  ```

- **RCX Interrupt table**
  ```c
typedef void (RCX_Handler)(void);
RCX_Handler ** RCX_Int_Table = (RCX_Handler **)0xfd80;
RCX_Int_Table[n] = &rcx_handler;
  ```
Hitachi H8/3292 I/O

- **7 I/O ports**
  - P1-P4, P6: 8-bit, input/output
  - P5: 3-bit, input/output
  - P7: 8-bit, input

- **Each comprised of up to three registers**
  - DDR (P1-P6): data direction (input/output)
  - DR (P1-P7): data
  - PCR (P1-P3): pull-up control

- **3 operating modes**
  - I: I/O address space, on-chip ROM disabled
  - II: I/O address space, on-chip ROM enabled
  - III: general I/O
## Hitachi H8/3292 I/O

<table>
<thead>
<tr>
<th>Port</th>
<th>Type</th>
<th>Mode I</th>
<th>Mode II</th>
<th>Mode III</th>
<th>DDR</th>
<th>DR</th>
<th>PCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8-bit, I/O</td>
<td>16-bit I/O address (lsb)</td>
<td>GPIO</td>
<td>0xffb0</td>
<td>0xffb2</td>
<td>0xffac</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>8-bit, I/O</td>
<td>16-bit I/O address (msb)</td>
<td>GPIO</td>
<td>0xffb1</td>
<td>0xffb3</td>
<td>0xffad</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>8-bit, I/O</td>
<td>8-bit I/O data</td>
<td>GPIO</td>
<td>0xffb4</td>
<td>0xffb6</td>
<td>0xffae</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>8-bit, I/O</td>
<td>bus state (IRQ/WAIT/RD/WR/CLOCK/AD)</td>
<td>GPIO</td>
<td>0xffb5</td>
<td>0xffb7</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3-bit, I/O</td>
<td>serial port</td>
<td></td>
<td>0xffb8</td>
<td>0xffba</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8-bit, I/O</td>
<td>timer control</td>
<td></td>
<td>0xffb9</td>
<td>0xffbb</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8-bit, I/O</td>
<td>A/D converter</td>
<td></td>
<td>–</td>
<td></td>
<td>0xffbe</td>
<td>–</td>
</tr>
</tbody>
</table>
Hitachi H8/3292 I/O

- **A/D converter**
  - Control/status register to initiate and monitor conversions
  - Can trigger interrupts

ADCSR at 0xffe8

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Abbreviation</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>A/D End Flag</td>
<td>ADF</td>
<td>1 = end of conversion (must be cleared)</td>
</tr>
<tr>
<td>6</td>
<td>A/D End Interrupt Enable</td>
<td>ADIE</td>
<td>0 = disabled; 1 = enabled</td>
</tr>
<tr>
<td>5</td>
<td>A/D Start</td>
<td>ADST</td>
<td>0 = stop; 1 = start</td>
</tr>
<tr>
<td>4</td>
<td>Scan Mode</td>
<td>SCAN</td>
<td>0 = single; 1 = scan</td>
</tr>
<tr>
<td>3</td>
<td>Clock Select</td>
<td>CKS</td>
<td>0 = slow; 1 = fast</td>
</tr>
<tr>
<td>2 - 0</td>
<td>Channel Select</td>
<td>CH2 - CH0</td>
<td>000 CH0, 001 CH1, 010 CH2, 011 CH3</td>
</tr>
</tbody>
</table>
Hitachi H8/3292 I/O

- A/D converter
- Three 10-bit channels (ADDR[A|B|C)]

**ADDR**

<table>
<thead>
<tr>
<th>Channel</th>
<th>Data Reg.</th>
<th>Addr.</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN0</td>
<td>A</td>
<td>0xffe0</td>
</tr>
<tr>
<td>AN1</td>
<td>B</td>
<td>0xffe2</td>
</tr>
<tr>
<td>AN2</td>
<td>C</td>
<td>0xffe4</td>
</tr>
</tbody>
</table>
LEGO RCX I/O

- H8/3292 I/O ports in operating mode II
- Buttons
  - 4 buttons connected to H8 I/O ports 4 and 7
  - Run and OnOff buttons can trigger interrupts

<table>
<thead>
<tr>
<th>RCX Button</th>
<th>H8 Port/Bit</th>
<th>H8 IRQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run</td>
<td>4/2</td>
<td>0</td>
</tr>
<tr>
<td>OnOff</td>
<td>4/1</td>
<td>1</td>
</tr>
<tr>
<td>View</td>
<td>7/6</td>
<td>-</td>
</tr>
<tr>
<td>Prgm</td>
<td>7/7</td>
<td>-</td>
</tr>
</tbody>
</table>
LEGO RCX I/O

- Sensors
  - 3 ports (1, 2, 3)
  - Connected to H8 I/O port 7 (A/D converter)
  - Passive (e.g. touch, temperature)
  - Active (e.g. light, rotation)
    - Activation via H8 I/O port 6

<table>
<thead>
<tr>
<th>RCX Sensor</th>
<th>H8 P7 bit (A/D ch)</th>
<th>H8 P6 bit (activate)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>AN2</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>AN1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>AN0</td>
<td>0</td>
</tr>
</tbody>
</table>
RCX Executive

- After a power up or reset, the H8/300 uses the first entry in the interrupt vector table (address 0x0000) to determine the starting address.
- In the RCX, this procedure activates an executive that:
  - Monitors the infrared port for application uploads.
  - Builds up a run-time environment for application programs, with services such as infrared communication, sensor and actuator manipulation, etc.