Wattbot:*nt*

Technical Notes

Jim Herd

March 2020

Contents

[Wattbot:*nt* 1](#_Toc67513308)

[1 Introduction 3](#_Toc67513309)

[2 System 3](#_Toc67513310)

[2.1 General system overview 3](#_Toc67513311)

[3 FPGA 4](#_Toc67513312)

[3.1 General FPGA overview 4](#_Toc67513313)

[4 Low-level microprocessor 5](#_Toc67513314)

[4.1 Data packets between microprocessor and FPGA. 5](#_Toc67513315)

[4.2 Process structure of uP software : V1 6](#_Toc67513316)

[5 High-level processor 7](#_Toc67513317)

Figures

[Figure 1 General system structure 3](file:///C:\jth\HW_new_robot\Wattbot_nt_technical.docx#_Toc33620361)

[Figure 2 General FPGA structure 4](file:///C:\jth\HW_new_robot\Wattbot_nt_technical.docx#_Toc33620362)

# 1 Introduction

This document brings together technical details relating to the design of Wattbot:*nt*.

# 2 System

## 2.1 General system overview

CAN bus

FPGA

H-bridge

(opt)

H-bridge

(opt)

H-bridge

(opt)

H-bridge

(opt)

Encoders

RC servo

RC servo

RC servo

RC servo

RC servo

RC servo

RC servo

RC servo

Low level

Microprocessor

High speed bus

Serial comms channel

High level

Control computer

e.g. Raspi

Real-time

clock

Dynamixel

servo bus

I2C

sensors

Motion

Control

System

H-bridge

(opt)

H-bridge

(opt)

H-bridge

(opt)

H-bridge

(opt)

External

H-bridge

Internal

H-bridge

DC motors

Figure 1 General system structure

# 3 FPGA

## 3.1 General FPGA overview

µP

FPGA

other functions

8-bit bus

32-bit bus

Low level

microprocessor

Digital I/O

FSM = Finite State Machine

Figure 2 General FPGA structure

# 4 Low-level microprocessor

# 4.1 Data packets between microprocessor and FPGA.

The data interface between the low level microcontroller is an 8-bit bidirectional bus. A state machine in the FPGA converts between this 8-bit bit and the internal 32-bit bus.

The microprocessor accesses the FPGA as a set of registers. The current system has 256 available registers (0-<255). At this time, only bits 0 and 7 of the command byte are used. Bit 0 is a READ/WRITE bit and bit 7 is a RESET bit. The 6-byte microprocessor to FPGA packet is as follows (shown in C struct format for documentation purposes)

**typedef** **struct** {

**cmd\_t** command;

**uint8\_t** register\_number;

**uint32\_t** data;

} uP\_to\_FPGA\_packet\_t;

**typedef** **struct** {

**int** R\_W :1;

**int** spare :6;

**int** reset :1;

} cmd\_t;

The 8-byte packet of data returned to the microprocessor consists of a data value and a status value, as follows

**typedef** **struct** {

**uint32\_t** data;

**uint32\_t** status;

} FPGA\_to\_uP\_packet\_t;

# 4.2 Process structure of uP software : V1

The uP runs the MBED RTOS software. The initial test structure uses the following structure with three processes (threads) and two mailbox queues.

FPGA

Com

Port

High level

controller

P3

P2

Low Level controller (µP)

P1

Queue1

Queue2

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Program name** | **Notes** |
| **P1** | Process | read\_from\_HLcontrol\_task | * Read ASCII command string from controlling computer * Send FPGA related commands to FPGA\_CMD\_queue * Implement non-FPGA commands |
| **P2** | Process | write\_to\_HLcontrol\_task | * Take ASCII reply srtings from reply queue and send to controlling computer |
| **P3** | Process | FPGA\_IO\_task | * Take FPGA command from FPGA command queue and |
| **Queue 1** | Mailbox | FPGA\_CMD\_QUEUE | * FIFO queue of LLcontrol (uP) commands coded as ASCII strings |
| **Queue 2** | Mailbox | HLcontrol\_reply\_queue | * FIFO queue of binary coded FPGA register commands |

This structure provides the basic system to test and exercise the FPGA hardware. The ASCII command format allows easy access from a high level control computer e.g. PC, Raspberry Pi, etc.

Testing uses a windows laptop with a C# program (March 2021).

# 5 High-level processor