Wattbot:*nt*

Technical Notes

Jim Herd

March 2020

Contents

[Wattbot:*nt* 1](#_Toc33737775)

[1 Introduction 3](#_Toc33737776)

[2 System 3](#_Toc33737777)

[2.1 General system overview 3](#_Toc33737778)

[3 FPGA 4](#_Toc33737779)

[3.1 General FPGA overview 4](#_Toc33737780)

[4 Low-level microprocessor 5](#_Toc33737781)

[4.1 Data packets between microprocessor and FPGA. 5](#_Toc33737782)

[5 High-level processor 6](#_Toc33737783)

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;

# 5 High-level processor