Using a Pre-Configured GoodnightPi v1.1

Note: These instructions assume that your Raspbian OS is already configured with the right repository and the appropriate additions to the crontab to allow the GoodnightPi to run. Instructions on setting up a GoodnightPi from scratch will be coming in the future.

Operation Overview

The GoodnightPi is intended to reduce the power consumption of the Raspberry Pi by allowing it to automatically turn on, run a program specified by the user, and then turn off again. The GoodnightPi can turn on a Pi based on a time interval or based on two different interrupts. The time of the interval or whether interrupts can be used can be set by the user using a simple GUI program included in the GoodnightPi repository. Status LEDs on the board indicate what wakeup modes are currently active.

The repository for all GoodnightPi code can be found on GitHub here:
https://github.com/fkummer/GoodnightPi

Board Switches

The three switches on the board must be set properly for correct operation. These switches control power to the system and change its operational mode. The GoodnightPi has two operational modes: Active mode, in which it controls power based on the timer and interrupts, and Configuration mode, in which power is left on so that the user can configure the system without fear of shutdown.

Switch 1, marked MAIN, controls power to the entire GoodnightPi board. When this switch is off the entire system will be off. When the switch is turned on, the ATMEGA that controls power to the Pi turns on. By default, when this switch is turned on the Raspberry Pi also turns on to allow for user configuration.

Switch 2, marked PI, allows for manual control of power to the Raspberry Pi. If Switch 1 is on and Switch 2 is on, the Raspberry Pi will be directly powered from the battery, allowing it to stay on no matter what the ATMEGA does. This is intended to be used when the user wants to use the Raspberry Pi normally and does not want power to be lost unexpectedly.

Switch 3, marked ACTIVE, allows the user to set the system to either Active mode or Configuration mode. In Active mode, the system will turn power to the Raspberry Pi on and off based on the settings of the timer and the interrupts. In Configuration mode, the Pi will not lose power so that the user can configure the system as desired. When the switch is On, the system is in Active mode. When the switch is off the system is in Configuration mode.

Figure 1 below outlines the switches in blue and shows the number of each switch.

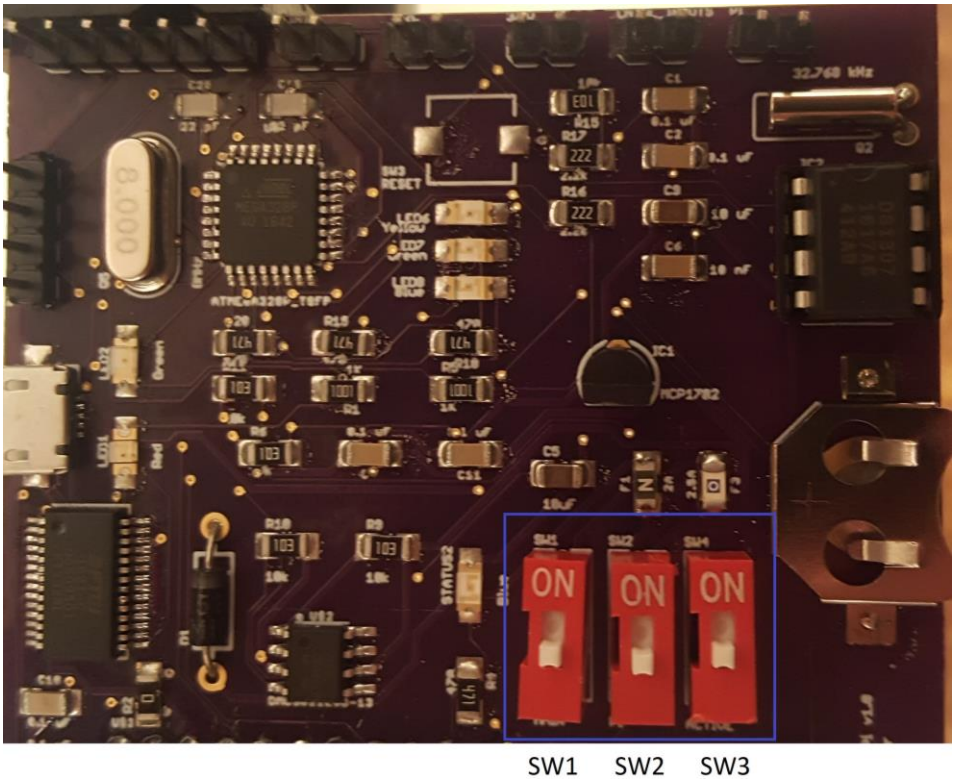

Figure 1: Location and Number of Board Switches

The sections below will provide the recommended switch configurations for the common ways the board would generally be employed.

Using the Raspberry Pi Normally

If you wish to use the Raspberry Pi normally, i.e. with no power control running, the switch configuration below should be used. This will set the system to Configuration mode and power the Pi directly.

Switch	Switch State
Switch 1	ON
Switch 2	ON
Switch 3	OFF

Using the System To Control Power

This is the primary mode the system is intended to run in, controlling power to the Pi based on timers and interrupts. The Pi is not directly powered, and the system is set to Active Mode.

Switch	Switch State
Switch 1	ON
Switch 2	OFF
Switch 3	ON

Configuring the System

These settings will allow the system to be configured, allowing the timers and interrupts to be set as desired and allowing the wakeup scripts to be changed as necessary. Note that to move the system back to Active Mode, first set switch 3 from OFF to ON, which will allow the Pi to shutdown gracefully, and then move switch 2 from ON to OFF, reverting to the switch settings used in the previous section "Using the System To Control Power".

Switch	Switch State
Switch 1	ON
Switch 2	ON
Switch 3	OFF

Configuring the System Using the GUI

The GUI can be found in the GoodnightPi repository. The gui2.py program should be executed using Python 2 to run the GUI. Note that PyGame is required to use the GUI, but on a pre-configured system this should already be included.

Once the GUI is open, it can be used to set the wakeup modes for the system. Using the arrow keys scrolls through different time intervals that the system can be set to wakeup on. Pressing enter will set the time interval. Clicking on either interrupt 0 or interrupt 1 will toggle whether a rising edge interrupt on the respective interrupt pin will trigger wakeup. To determine if the desired interrupt is currently on or off, refer to the status LEDs on the board. The Ping button can be used to determine if the SPI connection between the ATMEGA and the Pi is functioning properly. Pressing this button should result in a Ping Success message being displayed. To exit the GUI, simply click on the QUIT button.

Status LEDs

There are 3 status LEDs on the board that are currently used, LED6, LED7, and LED8. Each LED is used to indicate which wakeup modes are currently enabled. Note that the LEDs only turn on when the Raspberry Pi is on, most importantly during Configuration mode, in order to conserve power. LED6 turns on to indicate that Interrupt 0 is active, LED 7 turns on to indicate that Interrupt 1 is active, and LED8 turns on to indicate that the timer is active. Figure 2 below indicates the locations of the LEDs to help more easily identify them.

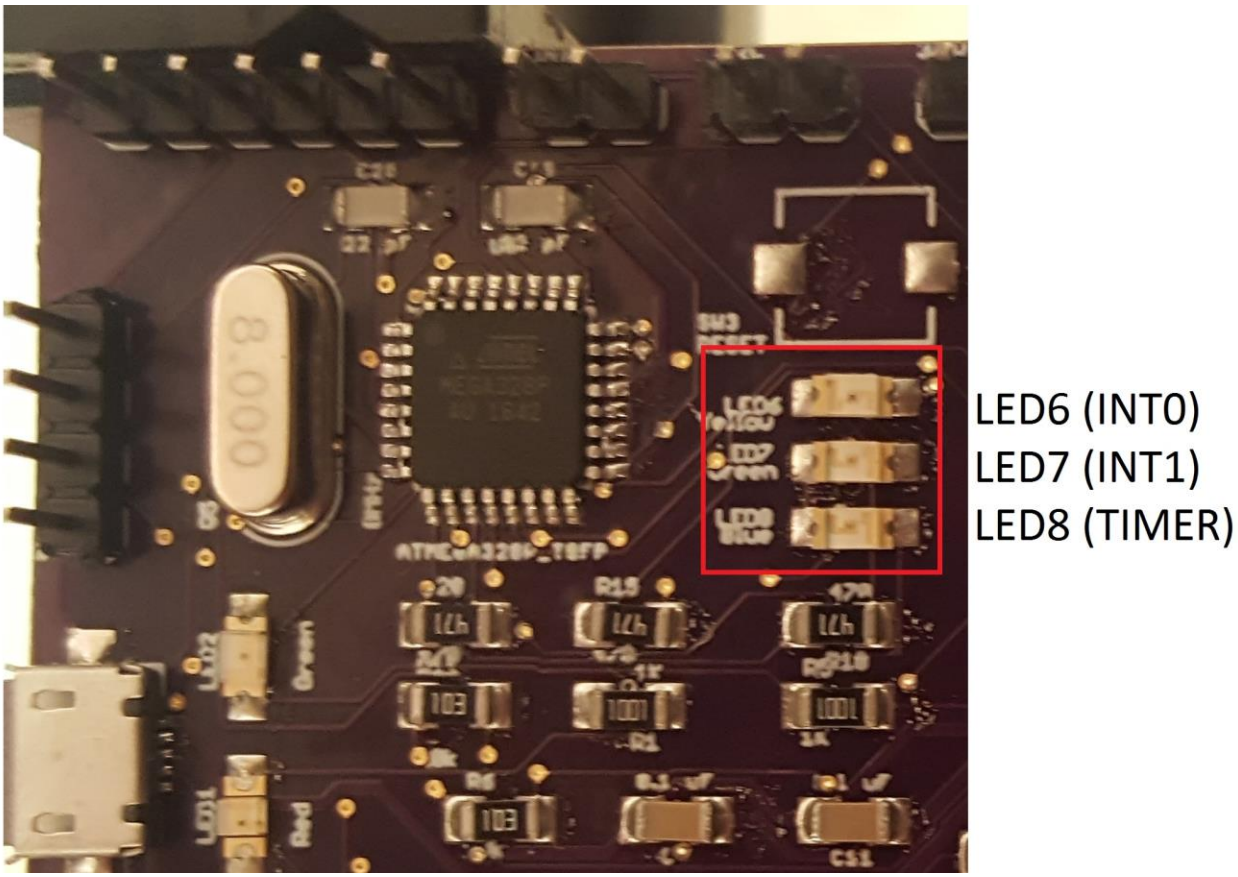

Figure 2: LED Locations and Numbers

Interrupts

Waking up of the Pi can be triggered by interrupts. Two different interrupts can be used, referred to as interrupt 0 and interrupt 1. The source of the interrupts is not important as long as the interrupts meet certain criteria. The interrupts must be based on 3.3V logic levels, with a minimum voltage of 0V and a maximum of 3.3V. The interrupts also must be rising interrupts, transitioning from low to high. Falling interrupts are currently not supported.

A two-pin male header is provided on the board to allow interrupts to be input to the system. This header is marked INT. Figure 3 below indicates the location of the INT header and which pin corresponds to which interrupt.

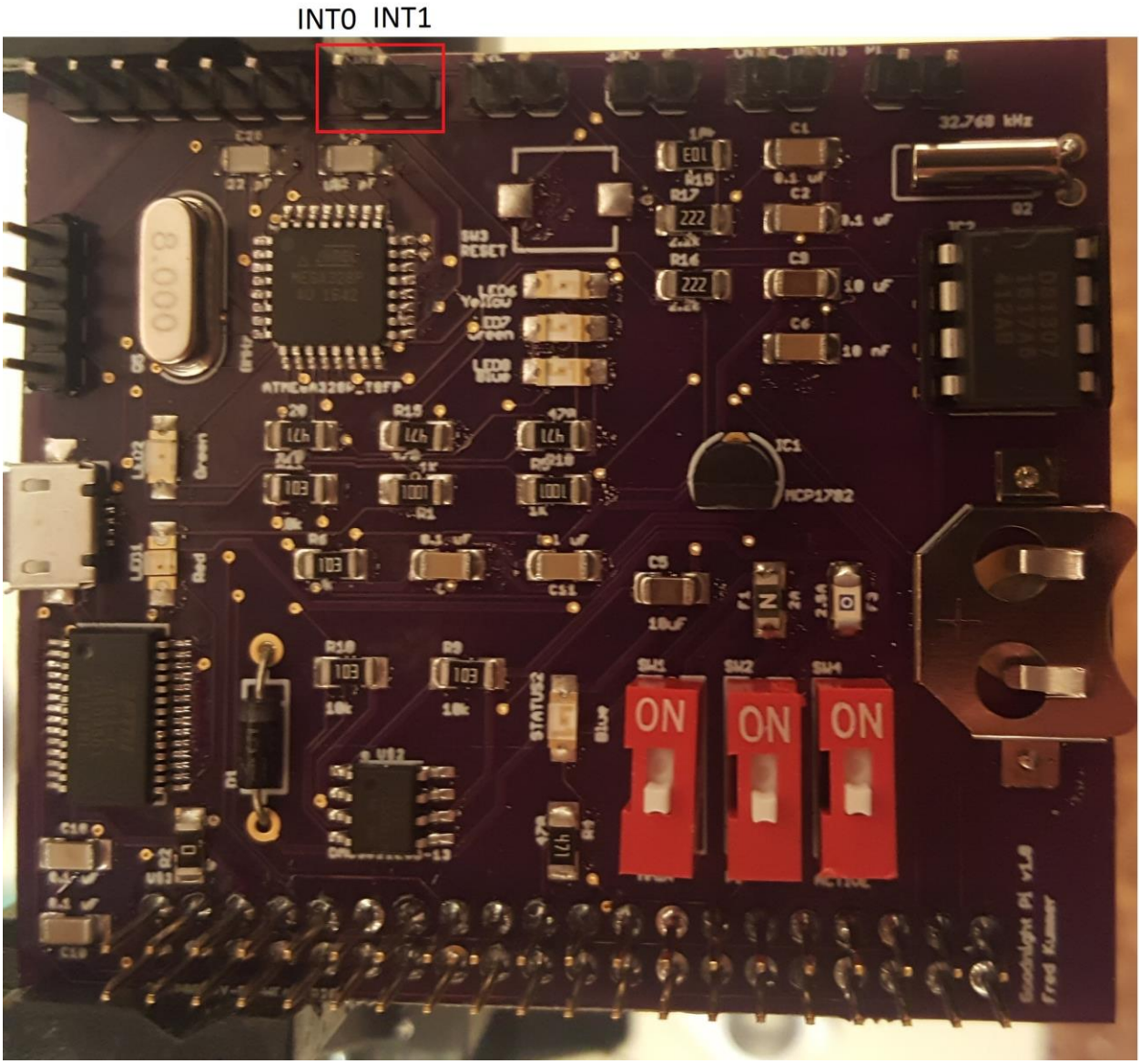

Figure 3: Location of Interrupt Headers

Wakeup Scripts

When the Raspberry Pi wakes up, it runs a python script based on the source that woke it up. So there are different scripts for waking up on a timer, interrupt 0, and interrupt 1. These scripts are all initially blank and must be defined by the user. They can be anything, and once they terminate the system will turn off again. The Timer.py, INT0.py, and INT1.py scripts in the GoodnightPi repository run on timer, interrupt 0, and interrupt 1 wakeups respectively. These should be edited by the user to run the desired functionality.