Project Objectives

In this project you will connect an LED to a Raspberry Pi's General-Purpose Input/Output (GPIO) header and control the LED using a cFS app. It allows cFS developers to gain experience with using a library and app to interface and control an external hardware component using low-cost materials.

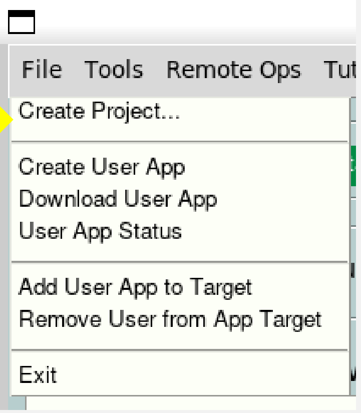
Detailed project instructions with videos can be found at

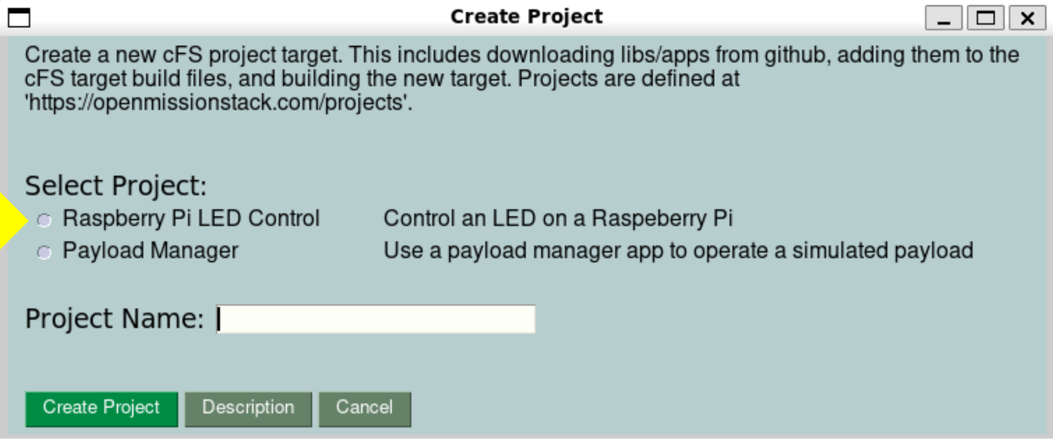
https://openmissionstack.com/projects_read/gpio_demo

UNDER CONSTRUCTION In fall 2024 projects will be rehosted to spacesteps.com

Sotware Installation

1. Create RPI LED Control project using the Create Project tool





Create a new cFS project target. This includes downloading libs/apps from github, adding them to the cFS target build files, and building the new target. Projects are defined at 'https://openmissionstack.com/projects'.

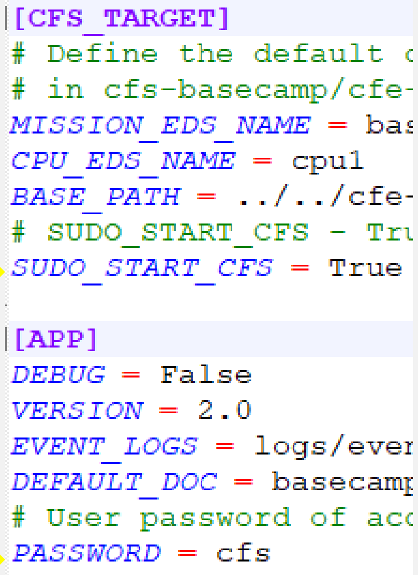
Select Project:

○ Raspberry Pi LED Control   Control an LED on a Raspeberry Pi

○ Payload Manager   Use a payload manager app to operate a simulated payload

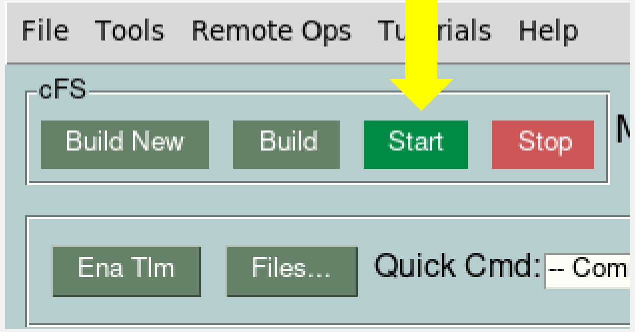
Project Name:

2. Before restarting Basecamp GUI, set basecamp.ini configurations to start the cFS with elevated permissions. The password needs to match your Raspberry Pi password



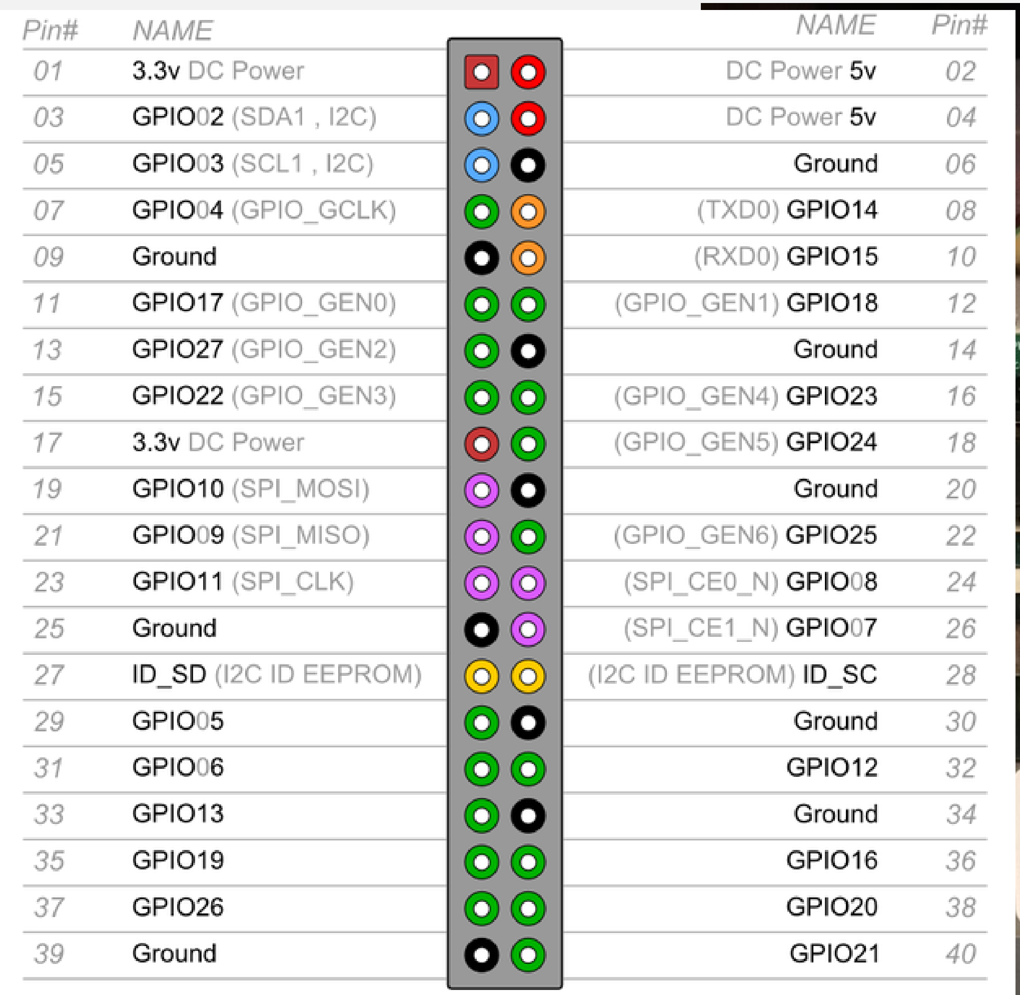
```
[CFS_TARGET]
# Define the default c
# in cfs-basecamp/cfe-
MISSION_EDS_NAME = bas
CPU_EDS_NAME = cpu1
BASE_PATH = ../../cfe-
# SUDO_START_CFS - Tru
SUDO_START_CFS = True

[APP]
DEBUG = False
VERSION = 2.0
EVENT_LOGS = logs/even
DEFAULT_DOC = basecamp
# User password of acc
PASSWORD = cfs
```

3. Start the cFS

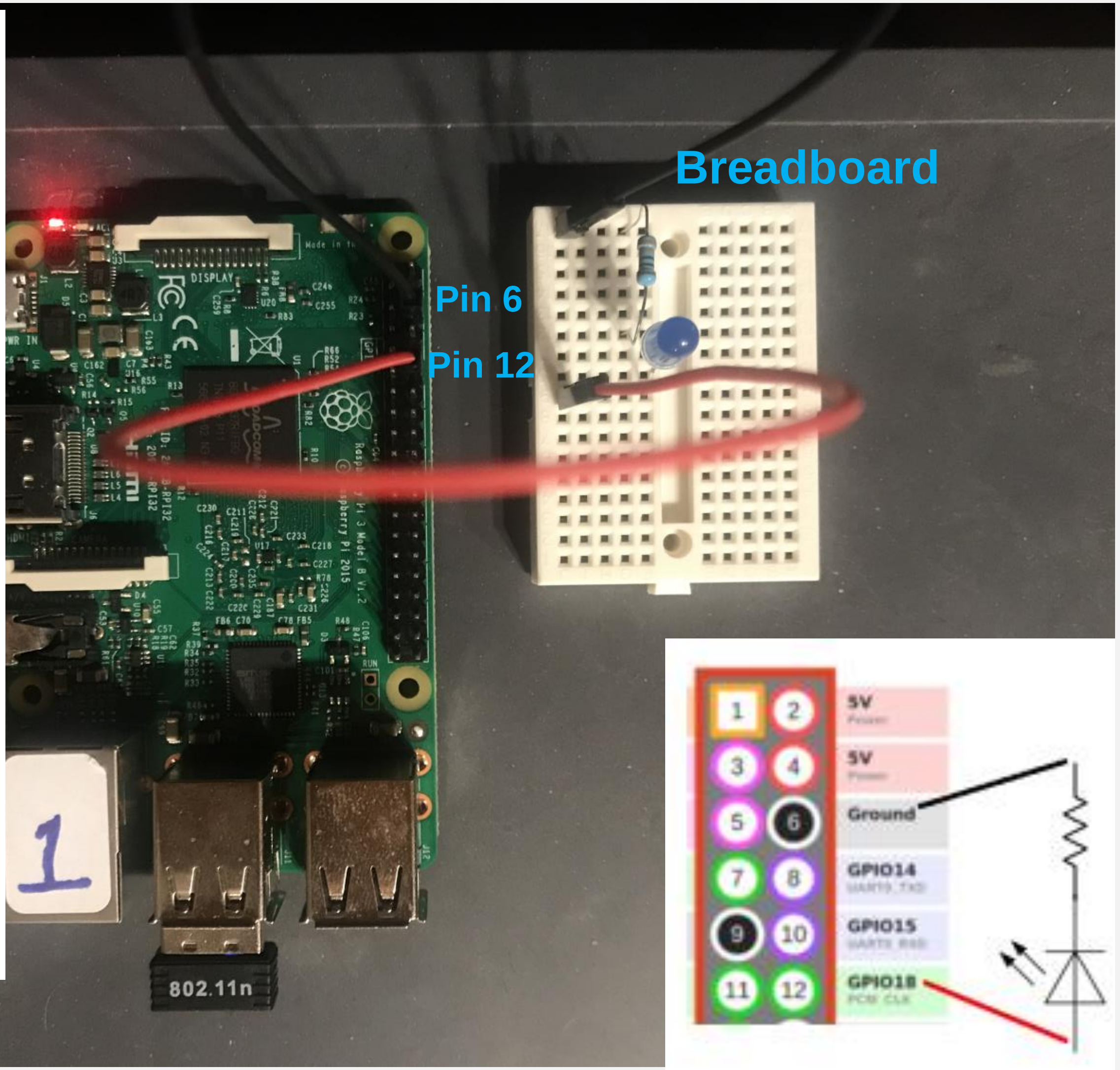


Hardware Assembly

1. Connect the resister and LED on a breadboard
2. Connect the breadboard to the GPIO pins as shown
   - Note physical pin 12 is logical GPIO pin 18 which is the identifier used by the FSW



Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1 , I2C)	DC Power 5v	04
05	GPIO03 (SCL1 , I2C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I2C ID EEPROM)	(I2C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

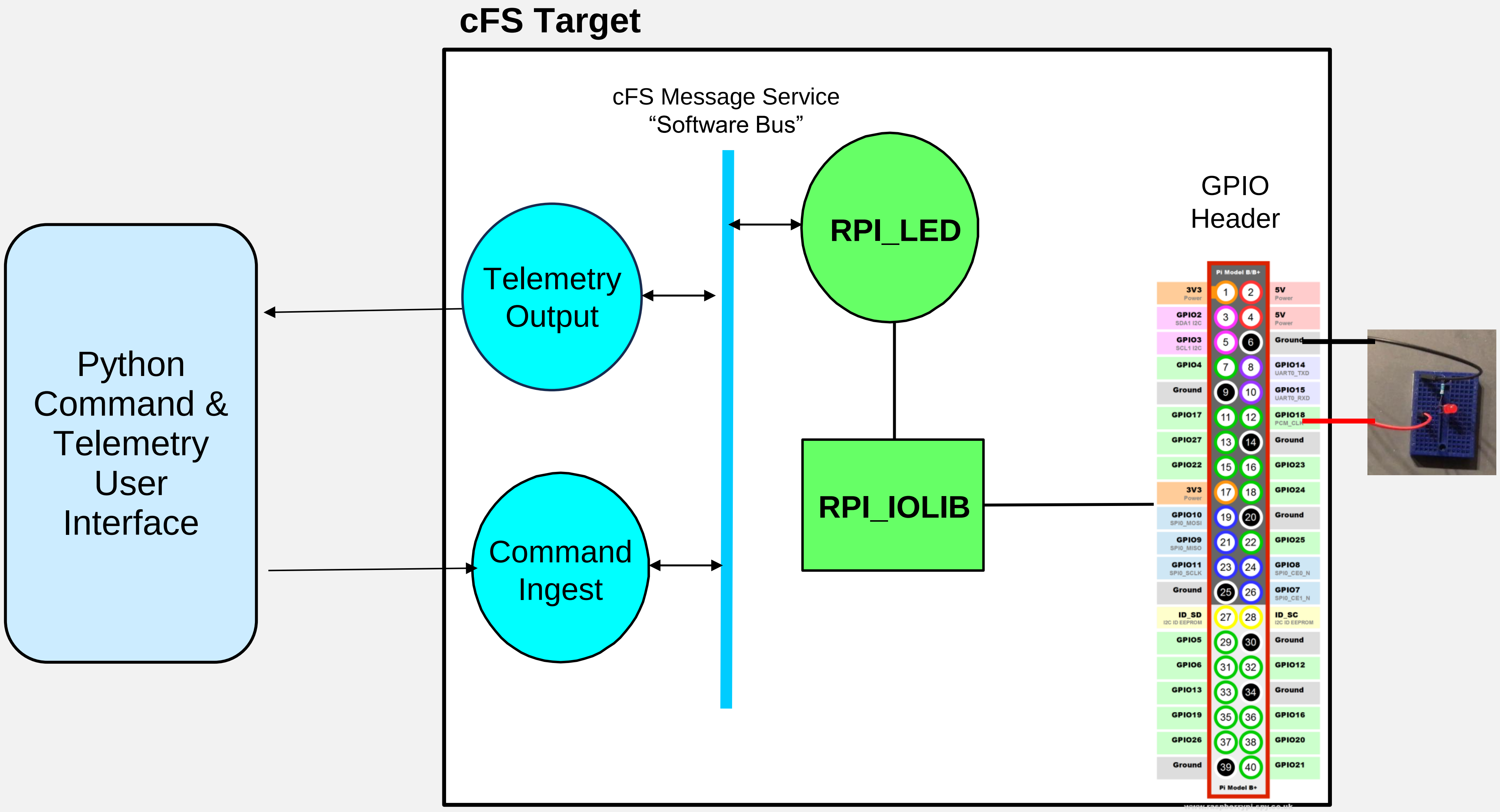


220 Ohm Resister
(red, red, brown)

LED

- *Connect shorter leg Cathode (-) to resistor*
- *Connect longer leg Anode (+) to GPIO pin 12*

Project Architecture



Pi Model B/B+

3V3 Power	1	2	5V Power
GPIO2 SDA1 I2C	3	4	5V Power
GPIO3 SCL1 I2C	5	6	Ground
GPIO4	7	8	GPIO14 UART0_TXD
Ground	9	10	GPIO15 UART0_RXD
GPIO17	11	12	GPIO18 PCM_CLK
GPIO27	13	14	Ground
GPIO22	15	16	GPIO23
3V3 Power	17	18	GPIO24
GPIO10 SPI0_MOSI	19	20	Ground
GPIO9 SPI0_MISO	21	22	GPIO25
GPIO11 SPI0_SCLK	23	24	GPIO8 SPI0_CE0_N
Ground	25	26	GPIO7 SPI0_CE1_N
ID_SD I2C ID EEPROM	27	28	ID_SC I2C ID EEPROM
GPIO5	29	30	Ground
GPIO6	31	32	GPIO12
GPIO13	33	34	Ground
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
Ground	39	40	GPIO21

Pi Model B+

www.raspberrypi-spy.co.uk

🟩 *= Project Apps and Library*

Project Library and App Summary

RPI_IOLIB

- This library adapts the “minimalistic peripheral access” (MIPEA) library for the Raspberry Pi, https://github.com/jasLogic/mipea so it can serve as a cFS library.
- It provides an interface to the Rasperry Pi’s memory mapped General Purpose I/O registers
- The config.h configuration file allows you to select your Broadcom processor chip

RPI_LED

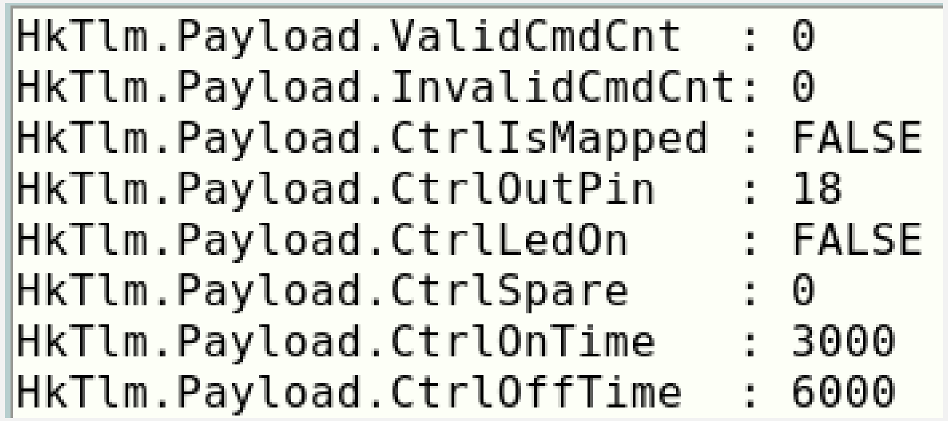
- The main app manages the ground command and telemetry interface
- A child task controls turning on and off the LED
- Ground commands can be sent to set the LED on and off time durations
- JSON initialization table defines the GPIO LED control pin and the default LED on/off time durations

RPI_LED App Operations

Commands

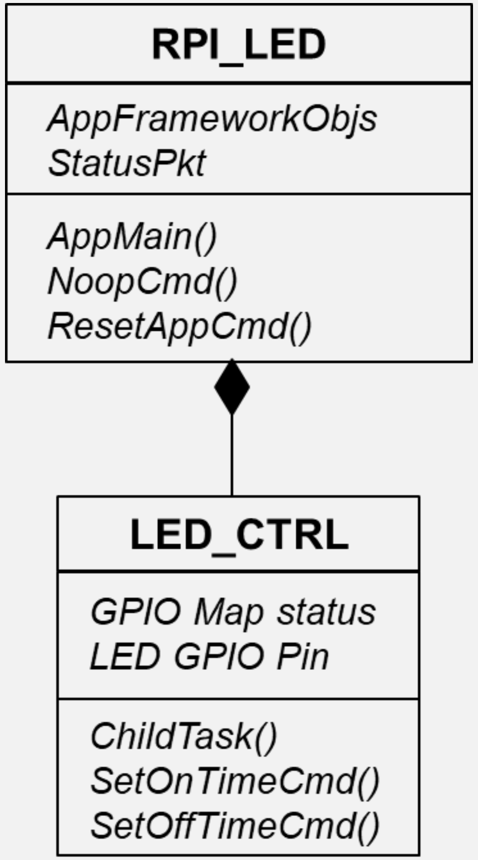
- Set LED On Time
- Set LED Off Time

Telemetry



```
HkTlm.Payload.ValidCmdCnt  : 0
HkTlm.Payload.InvalidCmdCnt: 0
HkTlm.Payload.CtrlIsMapped : FALSE
HkTlm.Payload.CtrlOutPin   : 18
HkTlm.Payload.CtrlLedOn    : FALSE
HkTlm.Payload.CtrlSpare    : 0
HkTlm.Payload.CtrlOnTime   : 3000
HkTlm.Payload.CtrlOffTime  : 6000
```

RPI_LED App Object Design



RPI_LED

AppFrameworkObjs
StatusPkt

AppMain()
NoopCmd()
ResetAppCmd()

LED_CTRL

GPIO Map status
LED GPIO Pin

ChildTask()
SetOnTimeCmd()
SetOffTimeCmd()

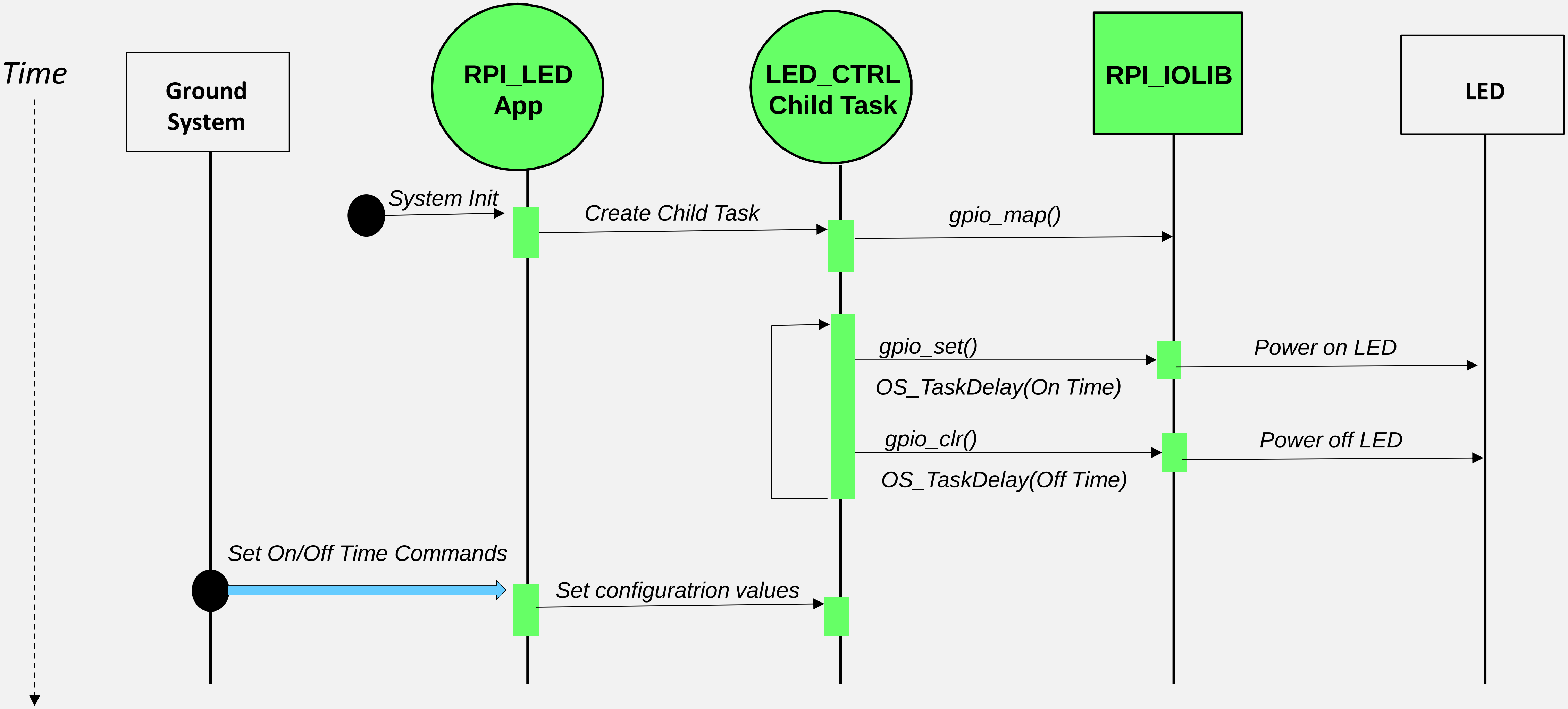
RPI_LED

- Manages app initialization, main runtime loop, and status telemetry
- Dispatches commands to objects

LED_CTRL

- Manages GPIO and LED interface
- Contains child task that turns the LED on and off
- Calls to RPI_IOLIB functions should be limited to this object to localize coupling

Control Flows



Time

Ground System

RPI_LED App

LED_CTRL Child Task

RPI_IOLIB

LED

System Init

Create Child Task

gpio_map()

gpio_set()

Power on LED

OS_TaskDelay(On Time)

gpio_clr()

Power off LED

OS_TaskDelay(Off Time)

Set On/Off Time Commands

Set configuratrion values

● Start Event

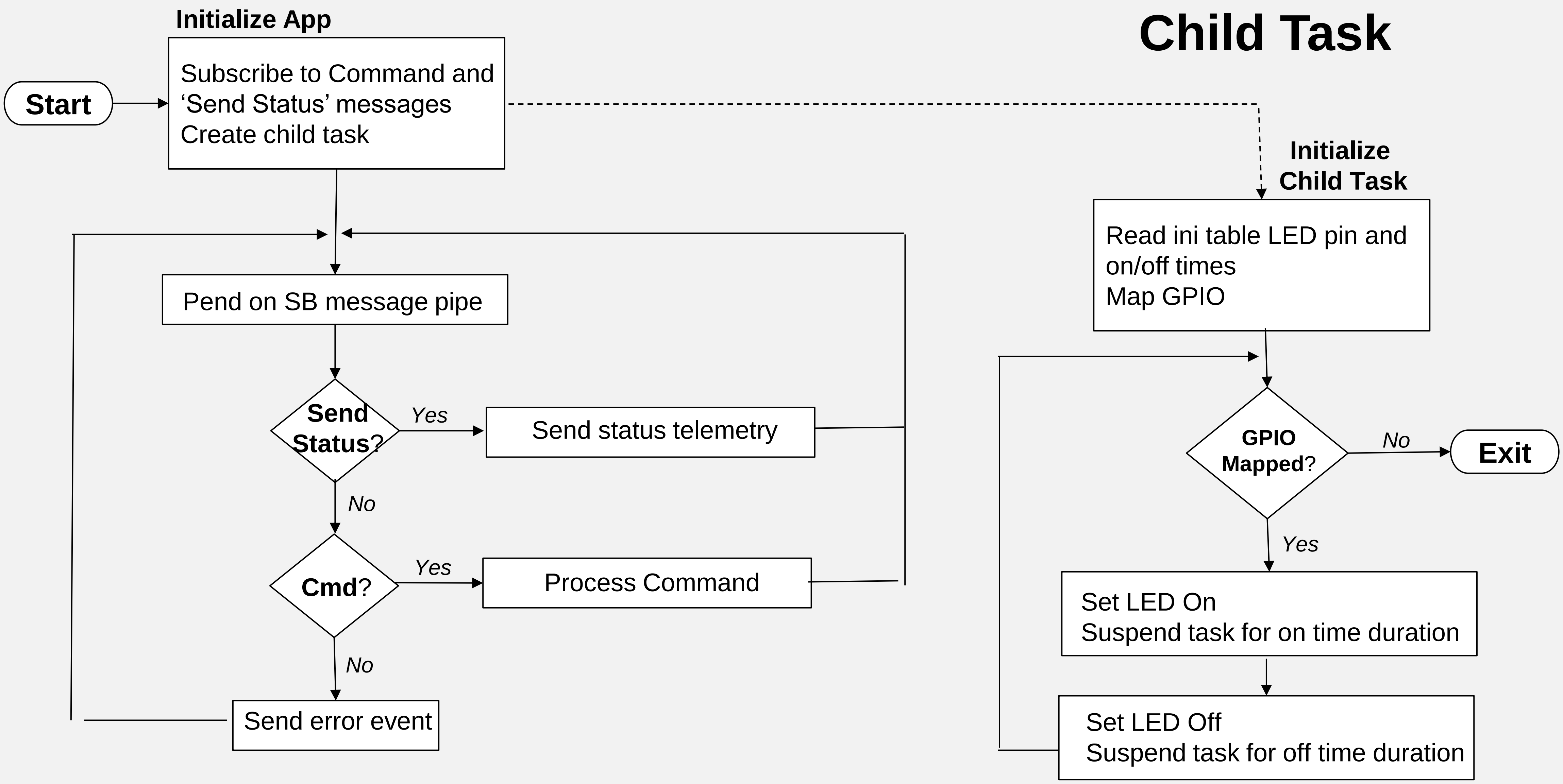
➡ Software Bus Message

→ Library Call

RPI_LED Control Flow

App

Child Task



Initialize App

Subscribe to Command and
‘Send Status’ messages
Create child task

Start

Pend on SB message pipe

Send Status?

Yes

Send status telemetry

No

Cmd?

Yes

Process Command

No

Send error event

Initialize Child Task

Read ini table LED pin and
on/off times
Map GPIO

GPIO Mapped?

No

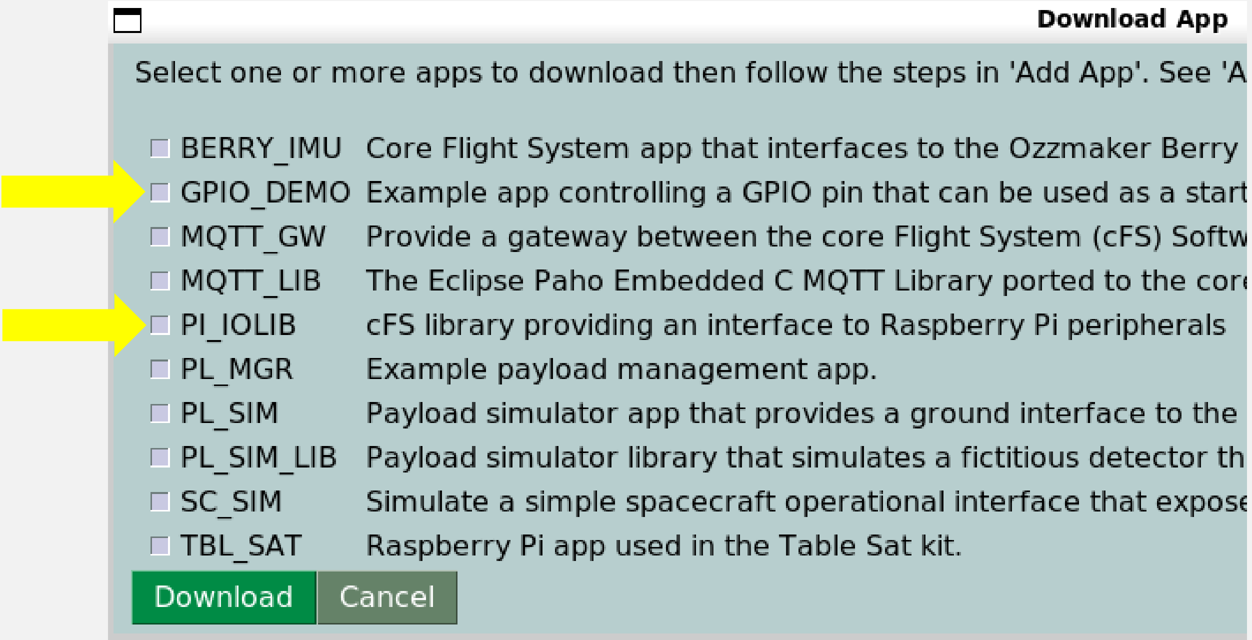
Exit

Yes

Set LED On
Suspend task for on time duration

Set LED Off
Suspend task for off time duration

Optional Manual Software Installation



Select one or more apps to download then follow the steps in 'Add App'. See 'A

□	BERRY_IMU	Core Flight System app that interfaces to the Ozzmaker Berry
□	GPIO_DEMO	Example app controlling a GPIO pin that can be used as a start
□	MQTT_GW	Provide a gateway between the core Flight System (cFS) Softw
□	MQTT_LIB	The Eclipse Paho Embedded C MQTT Library ported to the core
□	PI_IOLIB	cFS library providing an interface to Raspberry Pi peripherals
□	PL_MGR	Example payload management app.
□	PL_SIM	Payload simulator app that provides a ground interface to the
□	PL_SIM_LIB	Payload simulator library that simulates a fictitious detector th
□	SC_SIM	Simulate a simple spacecraft operational interface that expose
□	TBL_SAT	Raspberry Pi app used in the Table Sat kit.