VelaLab reference Manual

January 15, 2019

1 System Architecture

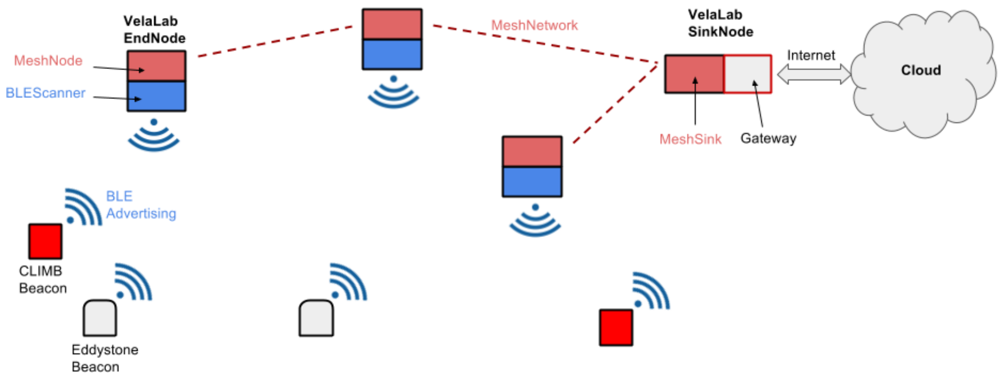

Figure 1:

1.1 Namings and definitions

BLE_Beacon

Any beacon device that can be detected from the system. Now it can be of two types: *CLIMB_Beacon* and *Eddystone_Beacon*.

VelaLab_EndNode

One of the nodes forming the BLE+mesh network. It is formed by a *BLE_Scanner* and a *Mesh_Node*, which communicate via a serial connection (UART).

BLE_Scanner

BLE part of an EndNode, based on the Nordinc nRFxx and implemented on the nRFxxx development board.

Mesh_Node

Mesh/Contiki part of an EndNode, based on the TI CCxxxxx and implemented on the Lauchpad xxx dev board.

VelaLab_SinkNode

The sink/gateway node of the network. It combines the Mesh_Sink and the Gateway, which are connected using the serial to USB feature on the Mesh_Sink. It does not have the BLE_Scanner (only mesh and gateway)

Mesh_Sink

The sink node for the mesh/contiki network. Its hardware is equivalent to a Mesh_Node, but it has a different firmware.

Gateway

The gateway between the network and the internet/cloud. It is implemented using a Raspberry PI 3 (RPI). The Mesh_Sink is connected through a USB port and it connects to the internet through Ethernet or WiFi.

Contact_Data Structure

Contact_Data is the data structure containing the information about the contact of one BLE Beacon. A *Report* is an array of *Contact_Data* containing all the data received during the last epoch. All the reported parameters refer to the last epoch.

Field	Type	Size (Byte)
`NodeID`	uint	6
`lastRSSI`	int	1
`maxRSSI`	int	1
`pktCounter`	u8	1

Back-End Data Structure (as in CLIMB)

Data structure containing the information to be sent from the gateway to the server, encoded as a json object. Each call send an array of such objects.

Field	Type
`wsnNodeId`	string
`eventType`	int
`timestamp`	int
`payload`	object

Payload may contain:

Field	Type
`passengerId`	string
`latitude`	float
`longitude`	float
`accuracy`	float

2 Beacons

Types:

- CLIMB (custom)
- Eddystone

3 VelaLab End-Node

Components:

- BLE Scanner
- Mesh Node

3.1 Hardware interconnection

fig...

The Launchpad image refers to the revision 1.0 of the hardware. In that version the DIO2 and DIO3 pins were incorrectly labelled (labels are swapped with respect to actual pin location). This scheme refers to the actual wiring position, then discard any label on the board and just wire as shown.

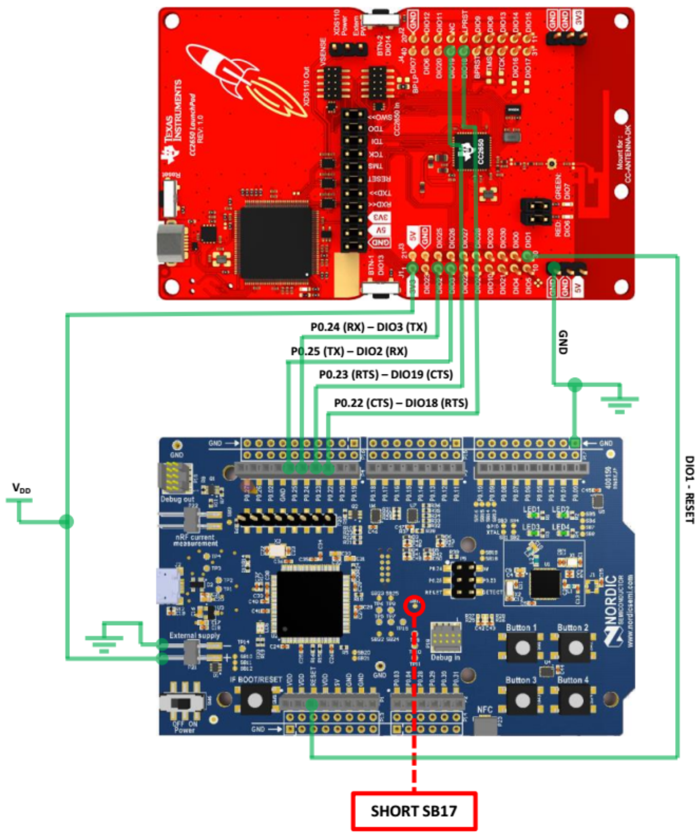

Figure 2:

3.2 Communication protocol

UART communication protocol between BLE Scanner and Mesh Node

4 VelaLab Sink-Node

Components:

- Mesh_Sink
- Gateway

4.1 Hardware interconnection

Launchpad usb output to RPI's USB

4.2 Communication protocol

UART communication protocol between Mesh Sink and Gateway

5 Mesh network

5.1 Mesh_Nodes to Sink-Node

The mesh network consists of atleast one Mesh_Node and a single Mesh_Sink. The destination of all the messages from the Mesh_Nodes is the Mesh_Sink. The Mesh_Nodes will send 3 types of messages to the Mesh_Sink: reports, keep alive messages, and battery data. Reports may get fragmented into fragments containing a maximum of 6 reports. Fragments only contain complete detections so that when a fragment gets lost the data from the other fragments can still be used. The Mesh_Sink will send all its data to the gateway for decoding and defragmentation. All the Mesh_Sink does is receive data and send it to the gateway without any checks or decoding etc.

5.2 Gateway to Mesh_Nodes

For configuring the Mesh_Nodes in the network the gateway can send messages into the mesh network through the Mesh_Sink. The following parameters can be configured: Enabling/disabling Bluetooth, Bluetooth duty cycle, keep alive messages interval, and battery info messages interval. The gateway sends a message to the Mesh_Sink and the Mesh_Sink will propegate a duplicate of this message through the mesh network using the Trickle protocol. This happens based on user input in the python script running on the gateway.