Информация для участников

Правила и заметки

1) Можно использовать девайсы которые вы принесете, главное помнить, если девайс прямой конкурент Intel Edison, то надо будет объяснить почему вы его использовали. К примеру, Intel Edison в своем базовом виде этого не умеет - т.е. у Edison нет YXZ.
2) Писать код надо уже сейчас, для этого не нужен сам девайс. В идеале если вы принесете 100% готовый код на хакатон. А на хакатоне мы его доведем до ума.
3) Вам будут активно помогать не только словом, но и делом. Если вы испытываете проблемы, то не надо сидеть на месте, надо сообщить нам - делать это можно уже сейчас.
4) Хакатон с ночевкой, кормить будем. Расписание в конце документа.
5) Приносить паяльные станции и любые другие опасные приборы нельзя! СБ такое не пропустит.
6) Приносите паспорт или документ удостоверяющий вашу личность.
7) Внимательно прочитайте данный документ!
8) По умолчанию пытается захватить 1GB памяти, что для edison много. Чтобы этого избежать надо, указывать box.cfg{slab_alloc_arena = 0.5 } (0.5 = 0.5 GB) или меньше.
9) Tarantool хранит слепок RAM на диске, я советую во время разработки удалять *.xlog, *.snap файлы, если вы поменяли схему.
10) Подпишитесь на изменения в https://github.com/dedok/tarantool-on-edison репозитории чтобы быть в курсе новых примеров!
11) Изучите/Попробуйте каждый пример: https://github.com/dedok/tarantool-on-edison/tree/master/examples

Tarantool

Документация: https://tarantool.org/doc/ru/

Документация(eng): https://tarantool.org/doc/

Ссылка на телеграм чат: https://telegram.me/joinchat/ABEGwD2KQS3ZzYLbHwIrpA
В чате сидят самые активные члены комьюнити, заходите, там отвечают на вопросы!

Установка в Облако или на ноутбук: https://tarantool.org/download.html

Пакет для Edison: https://github.com/dedok/tarantool-on-edison/blob/master/opkg-packages/tarantool_1.6.8.692_x86.ipk

NginX upstream: https://github.com/tarantool/nginx_upstream_module
Данный модуль поможет построить web/mobile API.

(!) Важно, если вам удобней в Облаке-ноутбуке использовать Python, php и т.п., то не надо себя мучать, найдите коннектор для вашего любимого языка, тут https://github.com/tarantool или в google
Тоже самое касается и Edison(!) - Tarantool можно использовать только как DB, очередь сообщения и т.п. Да будет менее эффективно, но Вам важна скорость разработки!

Установика ПО на Edison

Заходите на Edison через ssh или screen over USB(https://software.intel.com/en-us/setting-up-serial-terminal-on-system-with-linux), выполните след. действия.

```
$ git clone https://github.com/dedok/tarantool-on-edison
$ cd tarantool-on-edison/opkg-packages
$ opkg install cetic-6lbr-router_0.1_x86.ipk
$ opkg install tarantool_1.6.8.692_x86.ipk
$ opkg install tarantool-mqtt_0.1_x86.ipk
$ opkg install tarantool-mraa_0.1_x86.ipk
```

Архитектура

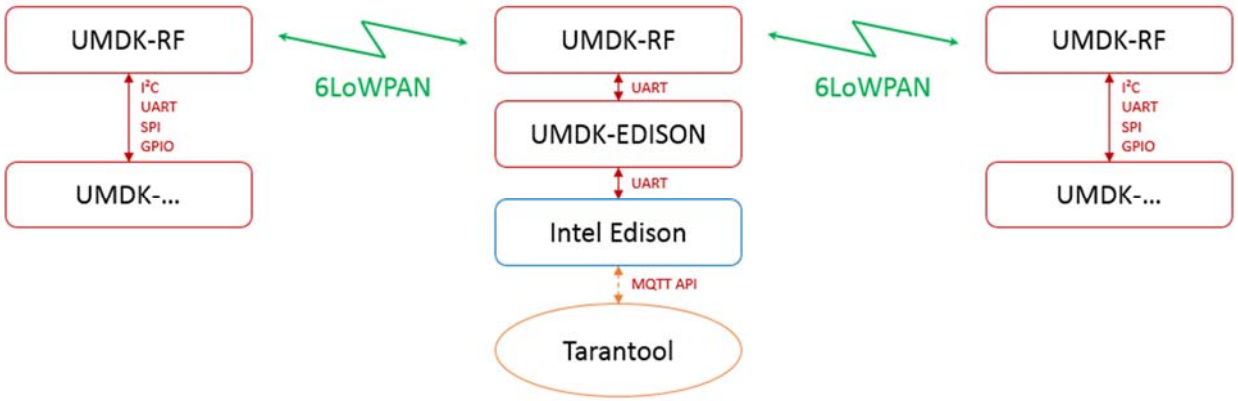
В начале, хочу отметить, Облако не означает Azure или Aws и т.п. Разворачивайте Tarantool у вас на ноутбуках(!) - это удобней и проще для разработки. Все мы (жюри) прекрасно понимаем - код который работает на ноутбуке легко перенести в Облако. Не мучайтесь разворачиваем инфраструктуры в Облаке!

А теперь к делу.

Unwired Devices

Так что такое Intel Edison - это, по меркам IoT, очень мощный девайс. Его задачи, в нашем понимании, сводятся к тому чтобы быть агрегатором, и обработчиком данных с более слабых представителей, к примеру, IoT-устройства с процессорами серии ARM cortex M. Следовательно можно выделить два класса устройств - слабые, и мощные. Встает вопрос как обеспечить взаимодействие всех этих устройств.

Ответ на этот вопрос предложила компания Unwired Devices(https://habrahabr.ru/company/mailru/blog/305506/): на Intel Edison стоит Tarantool и собирает данные через UMDK-RF, данные на UMDK-RF приходят через 6LoWPAN с более слабых устройств.



ПО для Edison и Облака/ноутбука

1) 6LoWPAN роутер - обмен сообщениями с девайсами через 6LoWPAN сеть, ссылка https://github.com/dedok/tarantool-on-edison/blob/master/opkg-packages/cetic-6lbr-router_0.1_x86.ipk

2) Tarantool - Lua application server и database, ссылка https://github.com/dedok/tarantool-on-edison/blob/master/opkg-packages/tarantool_1.6.8.692_x86.ipk

3) Tarantool MQTT - обмен сообщениями с MQTT брокером, ссылка https://github.com/dedok/tarantool-on-edison/blob/master/opkg-packages/tarantool-mqtt_0.1_x86.ipk

4) Примеры - https://github.com/dedok/tarantool-on-edison/tree/master/examples/UnwiredDevices

4.1) Пример работы с MQTT API - https://github.com/dedok/tarantool-on-edison/blob/master/examples/UnwiredDevices/6lbr_test.lua

5) Репликаций данных от 6LoWPAN

5.1) Серверная часть(Облако/Ноутбук) - https://github.com/dedok/tarantool-on-edison/blob/master/examples/UnwiredDevices/in_cloud.lua

5.2) Код для Edison - https://github.com/dedok/tarantool-on-edison/blob/master/examples/UnwiredDevices/in_device.lua

Описание MQTT API Unwired Devices

Брокер запускается в виде демона Mosquitto, собранного под MIPS.

- Файл конфигурации находится по пути /etc/mosquitto/mosquitto.conf
- Загружается автоматически

6LBR. Преобразование пакетов сети 6LoWPAN в вызовы MQTT выполняет демон 6LBR.

- Файл конфигурации находится по пути /etc/6lbr/6lbr.conf
- Загружается автоматически

Описание протокола

Корневой топик devices/, рекомендуется подписываться лучше на devices/# для получения всех сообщений по всем топикам.

Топик второго уровня, по-умолчанию Edison, может быть изменен на произвольный командой devices/Edison/topic_set. После исполнения этой команды следует заменить топик второго уровня Edison на новый пользовательский топик.
Значение пользовательского топика не сохраняется после перезагрузки.

Запрос списка подключенных устройств devices/Edison/get

Ответ на topic:devices/Edison/[IPv6 адрес устройства] в формате JSON message:

{{"ability": "N"}, {“last_seen":"3"}, {“last_sequience": "0"}, {"parent": “fe80::212:4b00:939:2d5d"}}

Поля

last_seen - время в секундах от прихода последнего сообщения со статусом от конечного устройства. Конечное устройство по-умолчанию присылает сообщение о статусе раз в 30 секунд.

last_sequence - номер последнего отправленного сообщения на устройство. Используется для формирования автоматического ответа от конечного устройства.

parent - IPv6 адрес устройства, через которое отправляет данные конечное устройство. Узел ближе к гейту по дереву маршрутизации mesh-сети.

ability - N определяет функциональность устройства. Каждая отдельная функция может быть определена наложением битовой маски на N., схема ниже.

Индекс	N	Функция
0	1	GPIO
1	2	PWM
2	3	BUTTON
3	4	GPIO_INPUT
4	5	ADC
5	6	DALI
6	7	SHT21
7	8	LPS331
8	9	OPT3001
9	10	LSM6DS3
10	11	INCOTEX
11	12	UART
12	13	A420

Для устройств, сочетающих несколько функций из списка выше, число N получается сложением.

IPv6-адрес устройства формируется из префикса сети (старшие 64 бита) и уникального EUI64-адреса устройства, который напечатан на модуле связи.
Пример ответного топика devices/Edison/aaaa::212:4b00:8fb:7e05 где aaaa:: префикс сети.

Команды и ответы устройств формируются из корневого топика, IPv6-адреса устройства и спецификатора команды/ответа.
Пример, devices/Edison/aaaa::212:4b00:8fb:7e05/gpio/n/toggle

Описание отдельных функций

GPIO

Устройство с такой прошивкой может управлять логическими выходами, которые могут быть подключены к реле (например UMDK-2RDC)

Команды:

Переключить состояние /gpio/n/toggle
Аргумент - ASCII строка с номером GPIO
Пример devices/Edison/aaaa::212:4b00:8fb:7d55/gpio/2/toggle 2
Номера выходов для UMDK-2RDC: ”1" - реле X1, ”2" - реле X2

Запрос состояния выхода /gpio/n/get
Аргумент может быть любой.
Пример devices/Edison/aaaa::212:4b00:8fb:7e05/gpio/1/get "1"
Ответ приходит на третий уровень топика /gpio_status
Пример:

devices/Edison/aaaa::212:4b00:8fb:7e05/gpio/status 1|0
devices/Edison/aaaa::212:4b00:8fb:7e05/gpio/status 2|1
Первое число - номер GPIO
Второе число - состояние выхода 0 или 1

Установить состояние /gpio/N/set L
Аргумент N — номер выхода 1 или 2
Аргумент L — состояние (0 или 1)
Пример devices/Edison/aaaa::212:4b00:8fb:7d55/gpio/2/set 1

PWM

Устройство с такой прошивкой может контролировать до 6 каналов ШИМ независимо.
Команды:

Добавить 20% скважности канала /pwm/N/toggle
N- номер канала от 1 до 6
Установить скважность /pwm/N/set D
N- номер канала от 1 до 6
Аргумент D — коэффициент заполнения (целое ASCII число от 0 до 99)
Узнать скважность /pwm/N/get 1
N - номер канала ШИМ
Ответ /pwm/status — коэффициент заполнения (0…99)

BUTTON
Устройство, работающее с кнопками (например UMDK-4BTN). При нажатии на кнопку присылает событие.

События кнопки приходит на топик /button/toggled
Пример devices/Edison/aaaa::212:4b00:8fb:7d55/button/toggled 1
Сообщение представляет собой ACII строку с номером нажатой кнопки от 1 до 4

ADC
Аналогово-Цифровой преобразователь.
Команды:

Установить номер входа /adc/set N
Аргумент - ASCII строка с номером входа от 24 до 30
Запрос напряжения на входе /adc/get N
Аргумент N - номер входа ADC от 24 до 30
При указании N вне пределов пинов 24-30 будет возвращено значение напряжения на заранее установленном входе командой /adc/set, либо с пина по умолчанию: 28
Значение напряжение на запрошенном пине публикуется на топик /adc/measure
Значение указано в микровольтах.
Минимальное 0(технически недостижимо, при подключении пада на землю существует разность потенциалов земли)

Максимальное 3.3В. Запрещается подавать на измерительный пин напряжение, превышающее это значение, относительно земли. Возможно повреждение процессора.
Пример: devices/Edison/aaaa::212:4b00:8fb:7d32/adc/measure 1580336

SHT21
Датчик температуры и влажности, установленный на плате UNDK-THP
Команды:

Запрос температуры окружающей среды `/sht21/temperature/get`
Запрос влажности воздуха `/sht21/humidity/get`
Сообщение должно быть не пустое

LPS331
Датчик давления, установленный на плате UNDK-THP
Команды:

Запрос давления /lps331/pressure/get

Сообщение должно быть не пустое

OPT3001
Датчик освещенности, установленный на плате UMDK-LIT
Команды:

Запрос освещенности в люменах на датчике /opt3001/light/get

Сообщение должно быть не пустое

UART
Доступ к последовательному асинхронному порту процессора. Может использоваться с платами расширения UMDK-RS485 и преобразователем в интерфейс RS232.
Команды:

Передача данных /uart/tx

Сообщение: данные, представленные в виде шестнадцатеричных ASCII символов. Например BEEF234E передаст в порт байты 0xBE, 0xEF, 0x23, 0x4E

Входные данные с порта конечного устройства приходят на топик /uart/rx в аналогичном виде

Второй тип архитектуры

Собрать прототип на Edison(ax) объединенных в кластер.
*кластер - репликация, или очереди сообщений (M-M Replication, MQTT).

Edison собирает данные, обрабатывает и т.п. Затем, обменивается данными с Облаком/ноутбуком.

ПО для Edison и Облака/ноутбука

1) Пример - https://github.com/dedok/tarantool-on-edison/tree/master/examples/security-system

1.1) Детальное описание - https://github.com/dedok/tarantool-on-edison/blob/master/examples/security-system/README.md

2) Tarantool - Lua application server и database, ссылка https://github.com/dedok/tarantool-on-edison/blob/master/opkg-packages/tarantool_1.6.8.692_x86.ipk

2) MRAA - для работы с PIN, SPI, UART и т.п., ссылка https://github.com/dedok/tarantool-on-edison/blob/master/opkg-packages/tarantool-mraa_0.1_x86.ipk

Расписание

Time Start	Time end	Presenter	Topic
День 1			
10:00	11:00		Welcome coffee и регистрация учасников
11:00	11:20		Вступление
11:20	13:30		Начало
13:30	14:15		Обед
14:15			
19:00	19:30		Ужин
19:30	-		
День 2			
10:00	10:30		Завтрак
10:30	14:00		
14:00	14:30		Обед
14:30	15:00		Финиш
15:00	17:30		Демонстрация
17:30	17:40		Отбор призовых мест
17:40	18:00		Финал