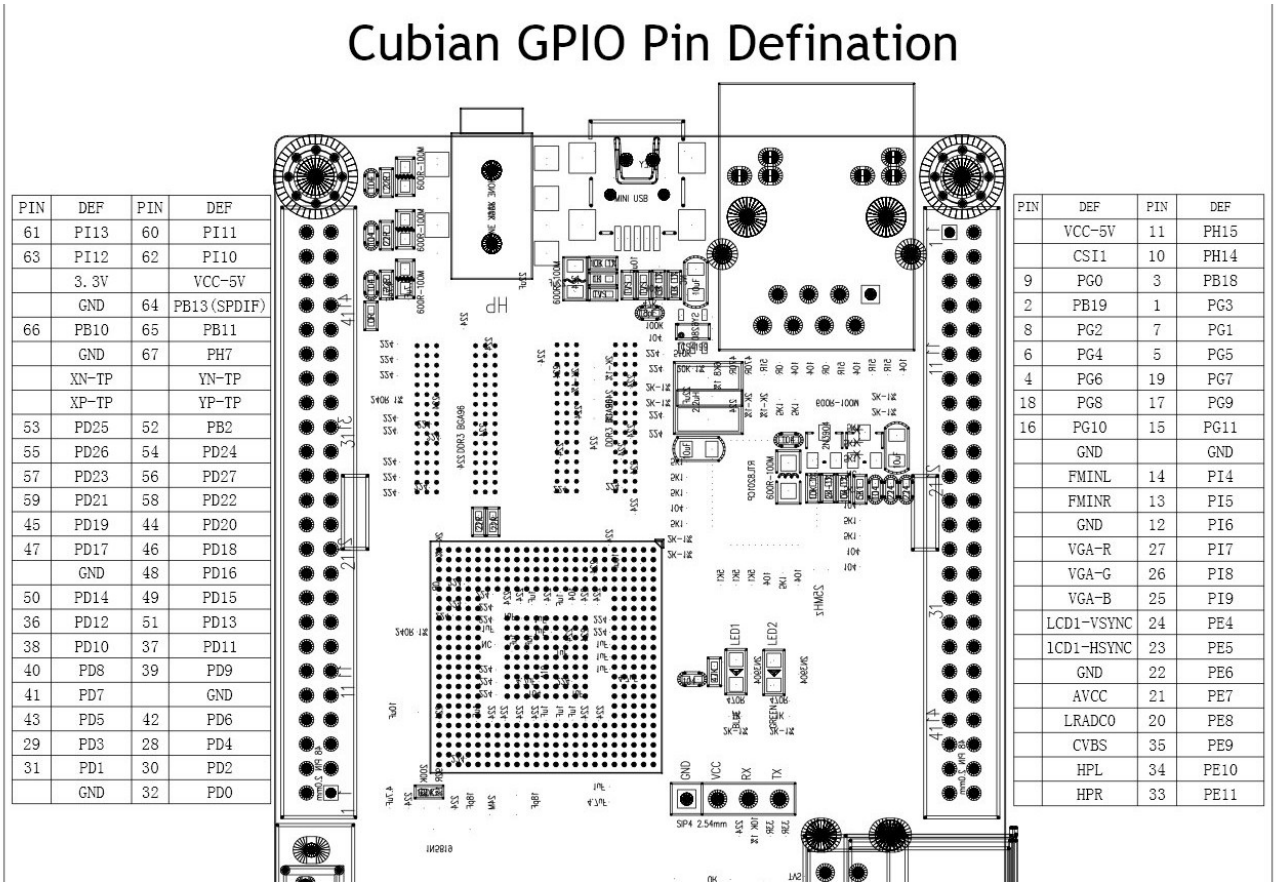
Использование GPIO на Cubieboard 1|2.

Степанов А. О.

Оглавление

0.1 Что такое GPIO.



PIN	DEF	PIN	DEF
61	PI13	60	PI11
63	PI12	62	PI10
	3.3V		VCC-5V
	GND	64	PB13(SPDIF)
66	PB10	65	PB11
	GND	67	PH7
	XN-TP		YN-TP
	XP-TP		YP-TP
53	PD25	52	PB2
55	PD26	54	PD24
57	PD23	56	PD27
59	PD21	58	PD22
45	PD19	44	PD20
47	PD17	46	PD18
	GND	48	PD16
50	PD14	49	PD15
36	PD12	51	PD13
38	PD10	37	PD11
40	PD8	39	PD9
41	PD7		GND
43	PD5	42	PD6
29	PD3	28	PD4
31	PD1	30	PD2
	GND	32	PD0

PIN	DEF	PIN	DEF
	VCC-5V	11	PH15
	CSI1	10	PH14
9	PG0	3	PB18
2	PB19	1	PG3
8	PG2	7	PG1
6	PG4	5	PG5
4	PG6	19	PG7
18	PG8	17	PG9
16	PG10	15	PG11
	GND		GND
	FMINL	14	PI4
	FMINR	13	PI5
	GND	12	PI6
	VGA-R	27	PI7
	VGA-G	26	PI8
	VGA-B	25	PI9
	LCD1-VSYNC	24	PE4
	LCD1-HSYNC	23	PE5
	GND	22	PE6
	AVCC	21	PE7
	LRADC0	20	PE8
	CVBS	35	PE9
	HPL	34	PE10
	HPR	33	PE11

Рис. 1: Номера и названия портов GPIO на плате

GPIO(General-purpose input/output) - это ряд специальных контактов общего назначения, пользователь может определить для каждого из них является ли он входом, либо выходом. [1] По умолчанию назначения не определены. Возможности GPIO:

1. может быть сконфигурировани для ввода или вывода
2. GPIO контакты могут быть отключены
3. на выходе может быть или 1 или 0
4. можно изменять выходные значения
5. часто можно использовать систему прерываний для GPIO

0.2 Особенности GPIO в Cubieboard 1|2.

Рассмотрите Рис. 1. Название линии GPIO имеет вид P{имя порта}{номер пота}. Графа PIN - это номер, по которому к нему можно обратиться из системы. Как видно из рисунка, не все линии можно испольовать для чего угодно, некоторые из них уже задействованы в конфиигурации системы. [2] Линии, помеченные как VCC-5V и 3.3V - это выводы для питания. Линии GND - земля. Здесь же распаяны интефейсы VGA, SPI и другие. Стоит обратить внимние, что растояние между штырками в Cubieboard гораздо меньше чем Paspberry PI или Arduino, это надо учитывать при выборе контактов для подключения.

0.3 Использование GPIO в ОС Linux.

0.3.1 Общие момоенты.

Прочитать подробно о GPIO в Linux можно в документе [LGIO00] GPIO в Linux после загрузки принадлежит системе, и для его использования каждую линию необходимо экспортировать. Для этого в каталоге */sys/class/gpio* в специальный файл *export* необходимо послать номер нужной линии. Например для открытия линии 17, которая отвечает порту G9 (PG9) Рис. 1: Код на bash:

```
# echo 17 > /sys/class/gpio/export
```

Код на C:

```
write(fd, "17", 1);
```

Если команда выполнилась успешно, то в дирректории */sys/class/gpio* появится дирректория *gpio17_pg9*. В ней есть файлы:

1. direction - определяет направление линии: является ли она входом или выходом, чтобы настроить линиюнужно записать в фал слово 'in' или 'out' соответственно. Например:

   ```
   # echo out > /sys/class/gpio/gpio17/direction
   ```

2. value - позволяет прочитать или считать значение с линии, в зависимости от того, какой тип в direction записан, in или out соответственно Например запись:

   ```
   # echo 1 > /sys/class/gpio/gpio17/value
   ```

   Или чтение:

   ```
   # cat /sys/class/gpio/gpio17/value
   ```

   Внимание! Это символы а не числа!

3. edge - управляет котроллером прерываний.

   (a) none - выключет отслеживание изменения состояния входящей линии;

   (b) rising - отслеживает переход из неактивного состояния в активное;

   (c) falling - отслеживает переход из активного состояния в неактивное;

   (d) both - реагирует на любые изменения состояния.

4. active_low - уровень активного сигнала, то есть напряжение логических нуля и единицы. По умолчанию в Cubieboard 1 - 3.3V, а 0 - 0V. Записанная сюда единица инвертирует активный сигнал

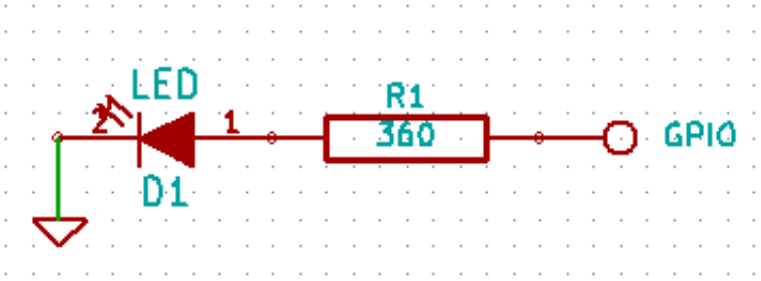


Рис. 2: Подключение светодиода к GPIO

0.3.2 Запись.

Возьмем простой пример, нам необходимо зажигать или выключать светодиод, подключенный по схеме Рис. 2 В качестве GPIO можно взять любой, пусть это будет 17. Включаем линию:

```
# echo 17 > /sys/class/gpio/export
```

Задаем ей направление на выход:

```
# echo out > /sys/class/gpio/gpio17_pg9/direction
```

Включаем 17-й, т. е. подаем на него напряжение:

```
# echo 1 > /sys/class/gpio/gpio17_pg9/value
```

Выключаем 17-й:

```
# echo 0 > /sys/class/gpio/gpio17_pg9/value
```

0.3.3 Чтение.

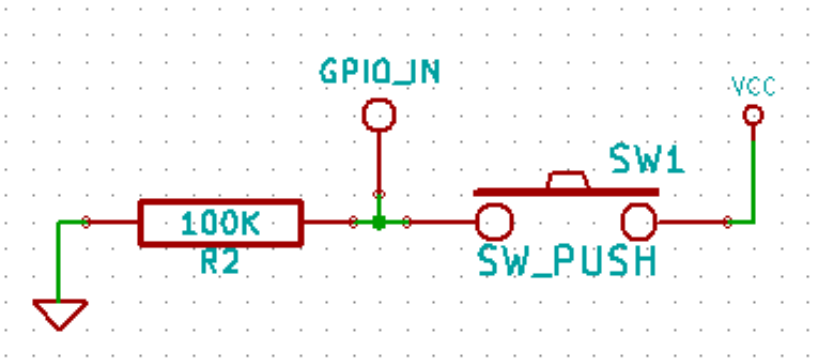


Рис. 3: Подключение светодиода к GPIO

Теперь чтение по схеме на Рис. 3: Аналогично в качестве *GPIO_IN* можно взять любой, пусть это опять будет 17. Включаем линию:

```
# echo 17 > /sys/class/gpio/export
```

Задаем ей направление на вход:

```
# echo in > /sys/class/gpio/gpio17_pg9/direction
```

Считываем значение:

```
# cat /sys/class/gpio/gpio17_pg9/value
```

Полученное значение зависит от *active_low*, если его не меняли, то мы получим символ(!) 1, если кнопка нпжата и 0 в противном случае. Соединение с землей нужно для предотвращения появления 1 на висящем в воздуже выводе.

0.3.4 Использование прерываний GPIO.

Для использования прерываний, необходимо активировать линию и записать способ слежения в *edge*. Повторюсь:

1. none - выключет отслеживание изменения состояния входящей линии;
2. rising - отслеживает переход из неактивного состояния в активное;
3. falling - отслеживает переход из активного состояния в неактивное;
4. both - реагирует на любые изменения состояния.

По инструкции, достаточно выбрать оди из способов кроме *none* и значение можно будет считывать с помощью *poll()* или *select()*. В случае, если значение изменилось, вызов *read()* должен быть заблокирован, но этого не произойдет. Так сделанно для работы команды *cat value*. [3] Из всего этого алгоритм звучит примерно так:

1. открыть файл *value*;
2. читать из него начальное значение(теперь файл будет блокироваться).

Есть еще одна тонкость: значения читаются только по смещению 0, в то время как вызов функции *read()* меняет позицию чтения. Поэтому позицию чтения необходимо сбромить с помощью *lseek()*. В итоге примерно так выглядит чтение GPIO с использованием событий *edge*:

```
// выбор способа отслеживания
int gpio_edge_set(int n, const char *edge_str)
{
    char filename[PATH_MAX];
    FILE *file;
    snprintf(filename, sizeof(filename), "/sys/class/gpio/gpio%d/edge", n);
    file = fopen(filename, "w");
    if (file == NULL) return -1;
    fprintf(file, "%s\n", edge_str);
    fclose(file);

    return 0;
}

// pool()
int gpio_poll(int n)
{
    char filename[PATH_MAX];
    int fd;
    char c;
    int err;

    snprintf(filename, sizeof(filename), "/sys/class/gpio/gpio%d/value", n);
    fd = open(filename, O_RDONLY);
    if (fd < 0) return -1;

    read(fd, &c, sizeof(c));

    return fd;
```

```
}

// получение значения с линии
int gpio_get(int fd, int timeout)
{
    struct pollfd pollfd[1];
    char c;
    int err;

    pollfd[0].fd = fd;
    pollfd[0].events = POLLPRI | POLLERR;
    pollfd[0].revents = 0;

    err =  poll(pollfd, 1, timeout);
    if(err != 1) return -1;

    lseek(fd, 0, SEEK_SET);
    err = read(fd, &c, sizeof(c));
    if(err != 1) return -1;

    return c - '0';
}
```

Литература

[1] Wikipedia. General-purpose input/output — Wikipedia, The Free Encyclopedia. 2014. [Online; accessed 7-March-2015]. URL: http://en.wikipedia.org/w/index.php?title=General-purpose_input/outputoldid=630541094.

[2] Cubian. GPIO Introduction. 2014. [Online; accessed 20-Mar-2014]. URL: https://github.com/cubieplayer/Cubian/wiki/GPIO-Introduction.

[3] scg. Linux: кнопки, светодиоды и GPIO. 2014. [Online; accessed 9-сентября-2014]. URL: http://habrahabr.ru/post/236251/.