CAN + Raspberry Pi

Update: An actual set of modules can be fouund here:Raspbian with actual 3.12.20+
Use at your own risk:

```
# update your Raspbian:
sudo apt-get update; sudo apt-get upgrade
# your kernel should be now: Linux raspberrypi 3.12.20+ #687 ...
# copy the tar archive to your RPi, e.g. /tmp
cd /tmp; wget http://lnxpps.de/rpie/rpi-can-3.12.20+.tar.bz2
# untar the archive:
cd /; sudo tar jxvf /tmp/rpi-can-3.12.20+.tar.bz2 # you normally should'nt do that
# register modules:
sudo depmod -a
# and do a
sudo reboot
# the spi-config set the config to the PICAN module
# load the module:
sudo modprobe mcp251x # or mcp2515
# setup the bitrate:
ip link set can0 up type can bitrate 500000
# watch dmesg for messages or stats:
ip -s -d link show can0
# ready to use with cansend, candump etc.
# be aware: the mcp2515 module is faster than the mcp251x but lacks the signalling:
#  you might see the status STOPPED even the CAN interface is still working fine
```

Compiled with the tz1 pikeromatic scripts (hash update to 3.12.20+).

Here is another way to compile the modules yourself : howto crosscompile modules Thanks to Damian Philipp for his description.

Please have a look at Raspberry Forum

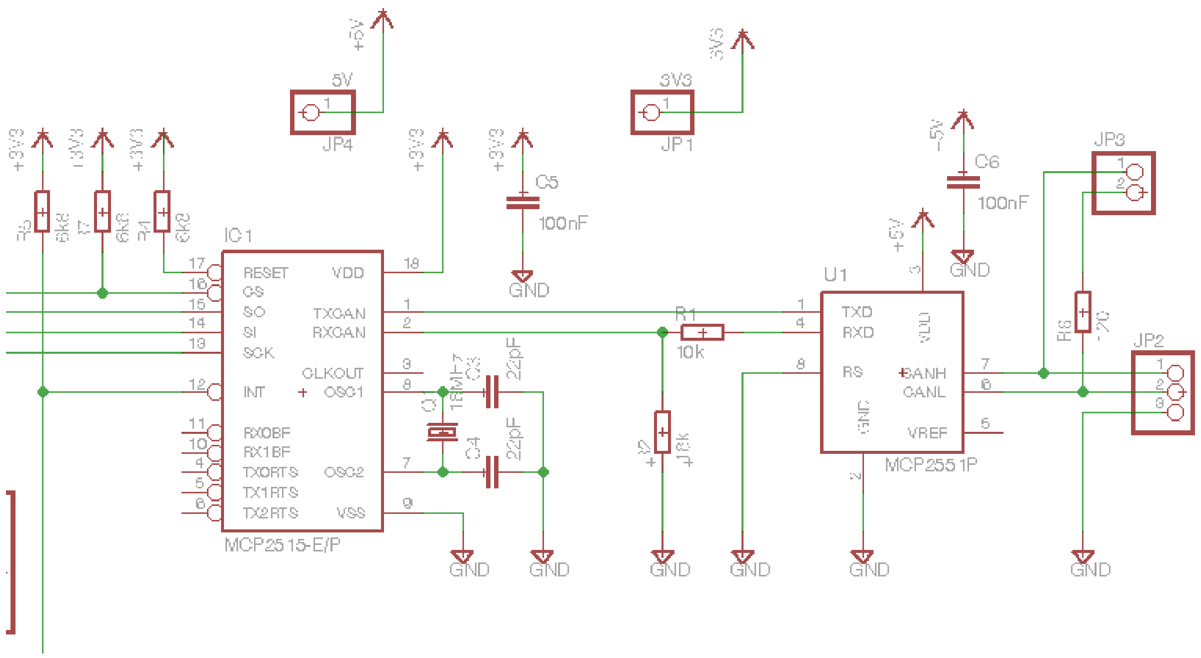
Summary

Efforts connecting a MCP2515 CAN controller to Raspberry Pi. Please note: This is not ment to be a description for Linux beginners. The combination of RPi and MCP2515 isn't perfect - you need some time to get a reliable setup working.

If you need fast and reliable CAN on a cheap SBC just use the BeagleBone Black.

Wiring

```
P1-01 3V3    -> MCP2515 VCC
P1-02 5V     -> MCP2551 VCC
P1-06 GND    -> MCP25xx GND
P1-19 GPIO10 -> MOSI
P1-21 GPIO9  -> MISO
P1-22 GPIO25 -> MCP2515 INT
P1-23 GPIO11 -> SCK
P1-24 GPIO8  -> CS0
```



Rasperry Pi GPIOs use 3V3 as the MCP2515 does. The tranceiver uses 5V - the R1/R2 combination is a voltage divider.

GPIO Test

Module to test GPIO (GPIO25) IRQ:
(Only used for testing -obsolete)

```
------------ gpio-test.c ---------------------8<-----------------------------------------
#include <linux/module.h>
#include <linux/init.h>
#include <linux/irq.h>
#include <linux/interrupt.h>
#include <linux/gpio.h>

int irq_number;

static irqreturn_t gpio_reset_interrupt(int irq, void* dev_id) {
        printk(KERN_ERR "gpio0 IRQ %d event",irq_number);
        return(IRQ_HANDLED);
}


static int __init mymodule_init(void) {
        irq_number = gpio_to_irq(25);

        if ( request_irq(irq_number, gpio_reset_interrupt, IRQF_TRIGGER_FALLING|IRQF_ONESHOT, "gpio_reset", NULL) ) {
                printk(KERN_ERR "GPIO_RESET: trouble requesting IRQ %d",irq_number);
                return(-EIO);
        } else {
                printk(KERN_ERR "GPIO_RESET: requesting IRQ %d-> fine\n",irq_number);
        }

        return 0;
}

static void __exit mymodule_exit(void) {
        free_irq(irq_number, NULL);
        printk ("gpio_reset module unloaded\n");
        return;
}

module_init(mymodule_init);
module_exit(mymodule_exit);

MODULE_LICENSE("GPL");

------------ Makefile ---------------------8<-----------------------------------------

obj-m += gpio-test.o
```

```
all:
	$(MAKE) -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules

clean:
	$(MAKE) -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```

Output after connecting to ground:

```
[ 4127.198958] GPIO_RESET: requesting IRQ 110-> fine
[ 4137.556627] gpio0 IRQ 110 event
...
[ 4168.574362] gpio0 IRQ 110 event
[ 4168.574397] gpio0 IRQ 110 event
[ 4168.574417] gpio0 IRQ 110 event
[ 4168.574434] gpio0 IRQ 110 event
[ 4168.574452] gpio0 IRQ 110 event
[ 4168.574477] gpio_reset module unloaded
```

The requested IRQ (here 110) may differ from kernel version used.

This module is only for testing the GPIO - don't use it when you want CAN !
All other modules are standard. They are already in the Kernel sources.

Board definition for kernel 3.2.27+

This is obsolete - use spi-config instead. The kernel doesn't need to be recompiled. spi-config from Martin Sperl changes the SPI kernel data structure.

```
/usr/src/linux-3.2.27+/arch/arm/mach-bcm2708/bcm2708.c

--- bcm2708.c_org   2012-09-21 14:33:39.102730640 +0200
+++ bcm2708.c   2012-09-21 20:19:09.560050330 +0200
@@ -54,6 +54,12 @@
 #include <mach/vcio.h>
 #include <mach/system.h>

+#include <linux/can/platform/mcp251x.h>
+#include <linux/gpio.h>
+#include <linux/irq.h>
+
+#define MCP2515_CAN_INT_GPIO_PIN 25
+
 #include "bcm2708.h"
 #include "armctrl.h"
 #include "clock.h"
@@ -579,10 +585,20 @@ static struct platform_device bcm2708_sp
    .resource = bcm2708_spi_resources,
 };

+static struct mcp251x_platform_data mcp251x_info = {
+   .oscillator_frequency   = 16000000,
+   .board_specific_setup   = NULL,
+   .irq_flags              = IRQF_TRIGGER_FALLING,
+   .power_enable           = NULL,
+   .transceiver_enable     = NULL,
+};
+
 static struct spi_board_info bcm2708_spi_devices[] = {
    {
-      .modalias = "spidev",
-      .max_speed_hz = 500000,
+      .modalias = "mcp2515",
+      .max_speed_hz = 10000000,
+      .platform_data = &mcp251x_info,
+      /* .irq = unknown , defined later thru bcm2708_mcp251x_init */
       .bus_num = 0,
       .chip_select = 0,
       .mode = SPI_MODE_0,
@@ -595,6 +611,12 @@ static struct spi_board_info bcm2708_spi
    }
 };

+static void __init bcm2708_mcp251x_init(void) {
+   bcm2708_spi_devices[0].irq = gpio_to_irq(MCP2515_CAN_INT_GPIO_PIN);
+   printk(KERN_INFO " BCM2708 mcp251x_init:  got IRQ %d for MCP2515\n", bcm2708_spi_devices[0].irq);
+   return;
+};
+
 static struct resource bcm2708_bsc0_resources[] = {
    {
       .start = BSC0_BASE,
@@ -723,6 +745,7 @@ void __init bcm2708_init(void)
    system_serial_low = serial;

 #ifdef CONFIG_SPI
+   bcm2708_mcp251x_init();
    spi_register_board_info(bcm2708_spi_devices,
          ARRAY_SIZE(bcm2708_spi_devices));
 #endif
```

For kernels > 3.6 you need:

```
+static struct mcp251x_platform_data mcp251x_info = {
+   .oscillator_frequency   = 16000000,
+   .board_specific_setup   = NULL,
+   .irq_flags              = IRQF_TRIGGER_FALLING|IRQF_ONESHOT,
+   .power_enable           = NULL,
+   .transceiver_enable     = NULL,
+};
```

People reported problems using ONE_SHOT mode - keep this in mind.

CAN test

Make sure that you have all necessary modules compiled and installed via 'make menuconfig; make'.

SocketCAN

libsocketcan
CAN utils

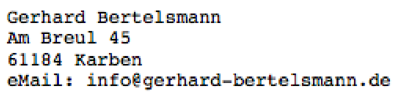
```
# initialize
insmod spi-bcm2708
insmod can
insmod can-dev
insmod can-raw
insmod can-bcm
insmod mcp251x
# Maerklin Gleisbox (60112 and 60113) uses 250000
# loopback mode for testing
ip link set can0 type can bitrate 125000 loopback on
ifconfig can0 up

root@raspberrypi ~ # dmesg
[  394.151290] bcm2708_spi bcm2708_spi.0: SPI Controller at 0x20204000 (irq 80)
[  465.325599] can: controller area network core (rev 20090105 abi 8)
[  465.325968] NET: Registered protocol family 29
[  523.007604] CAN device driver interface
[  560.310129] can: raw protocol (rev 20090105)
[  565.070666] can: broadcast manager protocol (rev 20090105 t)
[  593.259813] mcp251x spi0.0: CANSTAT 0x80 CANCTRL 0x07
[  593.266881] mcp251x spi0.0: probed
[  638.710821] mcp251x spi0.0: CNF: 0x03 0xb5 0x01

# on second terminal
root@raspberrypi ~ # candump any,0:0,#FFFFFFFF
  can0  123  [4] DE AD BE EF
  can0  123  [4] DE AD BE EF
  can0  123  [4] DE AD BE EF
  can0  123  [4] DE AD BE EF

root@raspberrypi ~ # cansend can0 123#deadbeef
root@raspberrypi ~ # cansend can0 123#deadbeef

root@raspberrypi ~ # ip -s -d link show can0
3: can0: <NOARP,UP,LOWER_UP,ECHO> mtu 16 qdisc pfifo_fast state UNKNOWN mode DEFAULT qlen 10
    link/can
    can <LOOPBACK> state ERROR-ACTIVE restart-ms 0
    bitrate 125000 sample-point 0.875
    tq 500 prop-seg 6 phase-seg1 7 phase-seg2 2 sjw 1
    mcp251x: tseg1 3..16 tseg2 2..8 sjw 1..4 brp 1..64 brp-inc 1
    clock 8000000
    re-started bus-errors arbit-lost error-warn error-pass bus-off
    0          0          0          0          0          0
    RX: bytes  packets  errors  dropped overrun mcast
    8          2        0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    8          2        0       0       0       0

root@raspberrypi ~# cat /proc/interrupts
           CPU0
  3:     192391   ARMCTRL  BCM2708 Timer Tick
 52:          2   ARMCTRL  BCM2708 GPIO catchall handler
 65:          2   ARMCTRL  ARM Mailbox IRQ
 66:          1   ARMCTRL  VCHIQ doorbell
 75:   14889016   ARMCTRL  dwc_otg, dwc_otg_hcd:usb1
 77:      11994   ARMCTRL  bcm2708_sdhci (dma)
 80:         58   ARMCTRL  bcm2708_spi.0
 83:         22   ARMCTRL  uart-pl011
 84:      21565   ARMCTRL  mmc0
110:          2      GPIO  mcp251x
Err:          0
```

Misc

Alternative MCP2515 module or modifed Team SocketCAN Version
Faster?!
Slightly modified:
Look at: Mail mcp2515.c

Reduced to the max

Impressum:



Gerhard Bertelsmann
Am Breul 45
61184 Karben
eMail: info@gerhard-bertelsmann.de