

Multifunctional **AR**duino dcc **DEC**oder



**the Multifunctional
DCC decoder for servo's
and accessory's
with DCCNext
for everybody**

Author: Nico Teering

January 2020

Mardec version: 6.0c

Document version: 6.0UK

Info: Info@Arcomora.com

www.Arcomora.com/Mardec/

Introduction

To keep the automation of your layout simple and cheap I created a program called Mardec for a multifunctional DCC decoder based on an Arduino processor.

With MARDEC servo's for turnouts as well as other accessories can be operated.

With MARDEC it is no longer necessary to write only one letter of Arduino code yourself.

As with any other DCC decoder, the MARDEC still needs to be configured.

This configuration is usually wrongly called programming. To avoid speech confusion we use the term programming to write the Arduino code and the term configure to run the program. In this configuration the servo's and accessories are set.

The MARDEC works in two different states .

In the **configuration mode**, the servos/accessories can be set by one-letter commands from the keyboard. The MARDEC communicates with the computer via a USB cable. No DCC signals are needed for this.

By means of a specific command ('E') the MARDEC switches to the **operating state**. (also called normal mode)

In the operating state, the servos and accessories are controlled with the DCC signal. No USB cable is needed anymore.

The installation of all software is very user-friendly. All required software is installed at once. Uploading the program to the decoder is also almost completely automatic.

See the installation manual for more information.

The decoder

For the decoder itself you have three options:

- An Arduino UNO or MEGA with the DCC/Power shield as described on <https://www.arcomora.com/mardec/>.
- An Arduino NANO with a homemade DCC circuit.
- The **DCCNext** decoder as described on <https://www.arcomora.com/dccnext>.

Mardec version 6 is made for the DCCNext but can also be used on an Arduino with DCC/Power shield.

ArCoMoRa

Mardec is part of the ArCoMoRa concept. It stands for Arduino Controlled Model Railway. The Arsigdec, a DCC signal decoder, is also part of this concept, as is the Arloco, a configurable LocoNet feedback system with Arduino.

See also: <https://www.Arcomora.com>

All MARDEC options at a glance

The MARDEC decoder has the following options:

- Driving a maximum of 12 servos without frogpoint polarization or a maximum of 8 servos with frogpoint polarization. The polarization is achieved through an external relay.
- Control of accessories in 9 ways, including control of alternating coils and variable PWM control.
- An Arduino port can be configured as input. This allows a servo / accessory on a different port to be switched on or off.
- Interactive, via screen and keyboard, configuring the inputs, servos and accessories. This is completely independent of the DCC central unit used. The Arduino software is NOT required for this.
- The start and end angles of each servo can be set precisely to the degree.
- Assign any DCC address (1-2000) to each servo or accessory. So not necessarily consecutive addresses.
- Each servo (max. 8) can be coupled with a relay for frogpoint polarization. When the servo is turned, this relay will be switched halfway through the rotation.
- Each servo (max. 5) can be coupled with two relays for frogpoint polarization. This results in an even more reliable switchover of the frogpoint.
- A different rotation speed can be set for each servo. This makes it easier for servos to be used for other purposes than just turnouts.
- A test option. Hereby the switching of the DCC-signal (0→ and 1→0) can be simulated.
- Servo's can bounce at the end of the rotation.
- Setting "inversion" for servos. With this, it is possible to set whether a switch should be set straight ahead or deflecting in the case of a rotation to the smallest corner. This is necessary because the servo can be mounted in several ways.
- A documentation option that shows all settings.
- Assigning an administrative code to each servo / accessory / input.
- A reset option where all settings are deleted from the memory of the MARDEC.
- Adjustable default speed of the servo arm (5-100 ms per degree).
- Return to configuration mode by connecting USB cable and entering "C" command.
- Possibility of correcting the address offset at Roco controllers (MM, z / Z21)
- Accessories have the following options:
 1. **Single steady.** Here, one port is permanently high or low depending on the control.
 2. **Double steady.** Same as single steady, but a second port has the "reverse" value.
 3. **Single flashing.** A port goes alternately high (on) and low (off). The "on" time and the "off time" can be set separately.
 4. **Double flashing.** Same as single double, but a second port has the "reverse" value.
 5. **Single one shot.** Here a gate goes as high for a short adjustable time as the control goes from low to high. It can also be used to switch a servo / accessory on another port at the end of the pulse.
 6. **Double one shot.** A port is set, for a short adjustable time, as "high" as the control goes from low to high and another port if it goes from high to low. This means that alternating coils can also be controlled.
 7. **Analog (PWM = Pulse Width Modulation) control.** In this case, in a separately adjustable time, a gate goes from 0 to an adjustable maximum (max. 255) if the control goes from low to high and then back to 0 in a separately adjustable time if the control goes from high to low.
 8. **Flicker mode.** With this you can make a connected LED flicker. With suitable LEDs you can, for example, simulate fire or a welding light.
 9. **Random.** This allows you to constantly switch a connected accessorie on and off. The on and off times are randomly determined between two adjustable limits (20 ms.-600 sec..) You can also set a fixed on or off time.
- A help option shows all commands for configuring.
- A log option that saves all configuration sessions.
- Independent of the used bus structure (LocoNet, S88 and the like). However, the control is only with DCC.

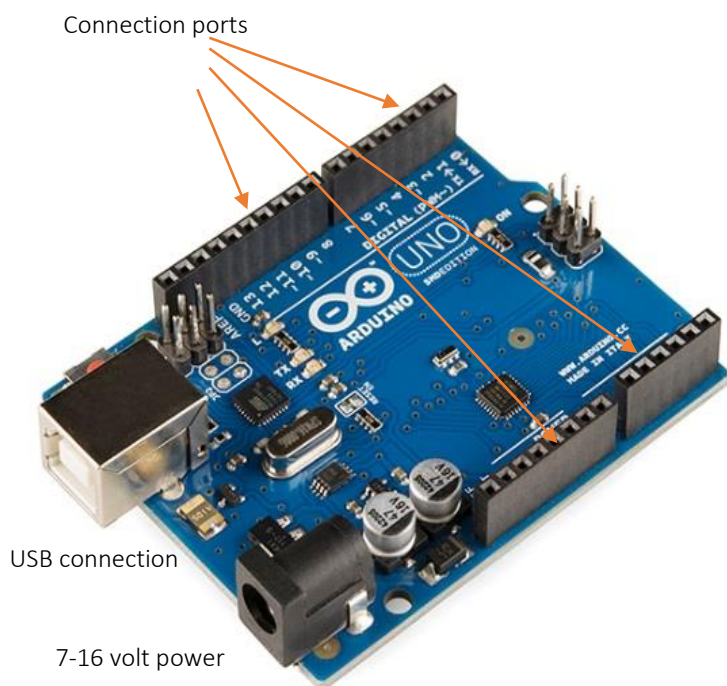
What do you need?

The hardware

You have a number of options here.

Option 1: An Arduino UNO, Arduino Mega2560 or Arduino Nano

An Arduino is a microcomputer with a lot of connections. These are called ports. They are numbered on the printed circuit board. We use a maximum of 16 of these ports for the MARDEC decoder. Servos, relay modules (see below) and LEDs (with resistor) can be directly connected to these ports.



An Arduino port has an output voltage 0 or 5 volts and can process a maximum of 40 mA. For accessories that require more power (such as motors, alternating coils and LED strips) an additional amplification stage is required.

For analogue applications there are 4 ports on which a block voltage can be set with a variable pulse width.

On the left you see the USB connection. The Arduino is powered by this USB cable.

The USB connection is required to copy the program to the Arduino. He is also needed to communicate with the computer when configuring.

If no USB cable is connected, the power must come via the other black connection socket (power jack). A voltage source of 7-16 Volt DC can be connected to this. The Arduino turns this into a stabilized 5 volt voltage. Both connections can be used at the same time.

On the Arduino we also see a number of ports with the text Power. Here we find two 5 volt ports (output), two GND

connections and a Vin port. Instead of the power jack, the Vin port can also be used to provide food to the Arduino. Do not connect 5V to the 5V ports.

DCC circuit

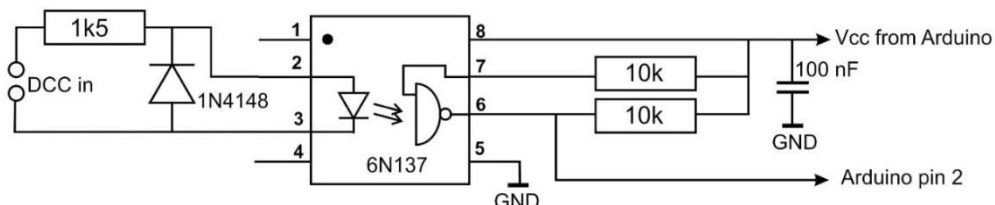
Secondly, a little bit of electronics is needed to make the DCC signal suitable for the Arduino.

The IC 6N137 is a fast optocoupler that transmits the DCC signal to the Arduino

The required 5 volts can be connected directly to a 5V port on the Arduino.

Port 6 of the IC is connected to port 2 of the Arduino.

You have to solder these extra components yourself on a mounting board.



DCC/Powershield

An add-on board (shield) has also been developed for this circuit. The power supply for the Arduino can also be placed on this. The print also contains a 5V power supply (max. 1 Amp) that can be used, among other things, for relays, LEDs and servos. However, an external power supply is recommended for servos and relays

This print can be ordered with the order form on:

<http://www.Arcomora.com/order>.

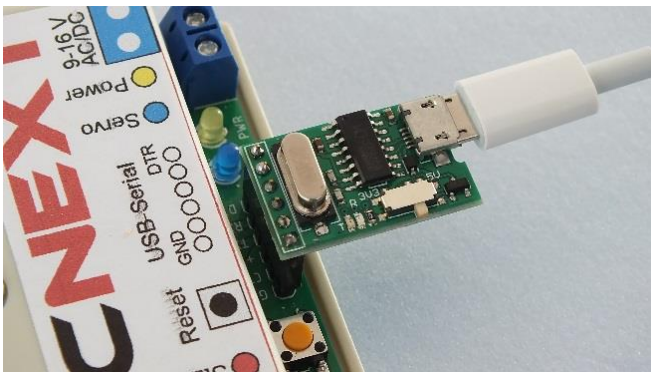
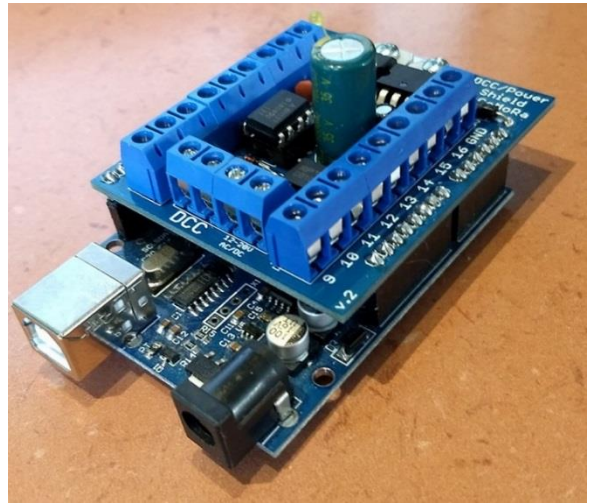
If not in stock, the delivery time can be a number of weeks. You also have to purchase an Arduino yourself

Option 2: The DCCNext decoder

Completely new is the DCCNext decoder.

This decoder integrates an Arduino processor (ATMEGA328P) with a power supply and a DCC circuit and is therefore a combination of an Arduino UNO and the DCC shield.

A separate USB interface (CH340) provides the connection to the PC.



This decoder can be equipped with both screw terminals and Dupont ports. A servo can be connected directly to the ports. A separate 5V servo power supply is also provided for this.

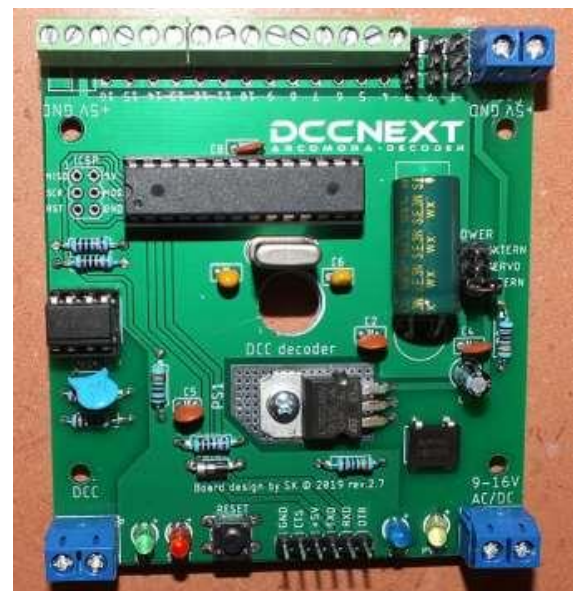
The presence of a DCC signal is made visible with a LED.

An important difference with the DCC shield is the use of the port numbers 1 to 16 instead of the port numbers of the Arduino (3 to 19). This manual assumes the port numbers of the DCCNext.

More info: <http://www.Arcomora.com/DCCNext>.

This DCCNext can be ordered with the order form on:

www.Arcomora.com/reservation.



Important:

Mardec 4 can also be used on the **DCCNext**.

Mardec 5 and 6 can also be used with the DCC shield.

In both cases the conversion table below applies:

Mardec 4 Pin	Mardec 5/6 Port	Mardec 4 Pin(Mega)	Mardec 5/6 Port
3	1	11	9
4	2	12	10
5	3	14(54)	11
6	4	15(55)	12
7	5	16(56)	13
8	6	17(57)	14
9	7	18(58)	15
10	8	19(59)	16

Mardec 5 and 6 uses a completely changed data structure for storing the configuration.

Mardec 4 can therefore NOT be updated to version 5.

If you replace Mardec 4 with Mardec 6 you have to reconfigure everything. Therefore, first make a screenshot of the current configuration before you install version 6 over version 4.

Relay

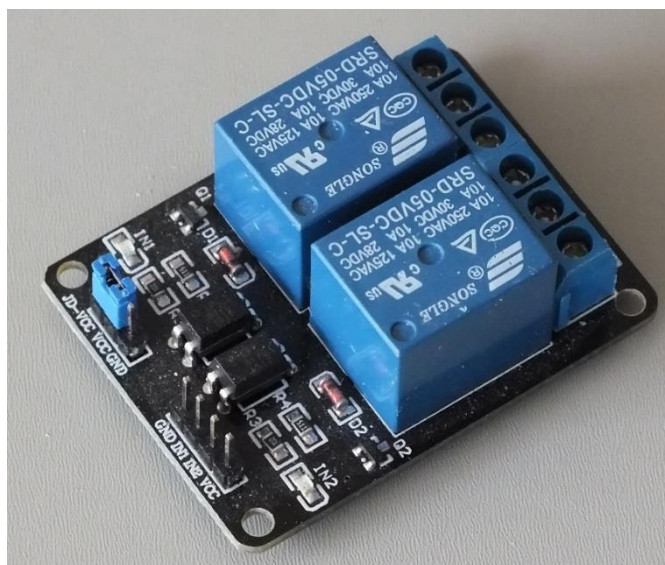
For the frogpoint polarization, or any other application, any relay can in principle be used.

To this end, an Arduinoport can switch a relay in the usual manner by means of a transistor.

[The ready-made relay modules](#) are cheaper and easier. These are very suitable for direct control from an Arduino.

They contain two independent relay circuits with an indication LED and an optocoupler and are also available with 4 or 8 relays.

They can be powered directly from the Arduino, but it is strongly recommended to use an external 5V power supply for this.



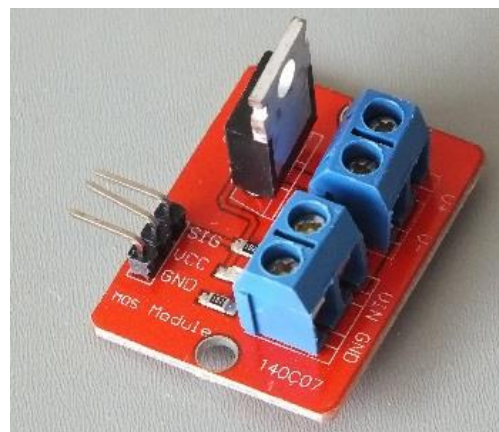
MOSFET switch

Switching is also possible with a [MOSFET amplifier](#).

They are suitable for alternating coils and LED strips, among other things.

They can also be used to dim with pulse width modulation (PWM). You can use a ready-made module for this or solder them yourself on a mounting PCB.

Suitable MOSFET types include the FQP30N06L and the RFP30N06LE. The "L" stands for "logic level". This means that these MOSFETs are fully conductive with a control voltage of 5V and have virtually no resistance anymore



Software

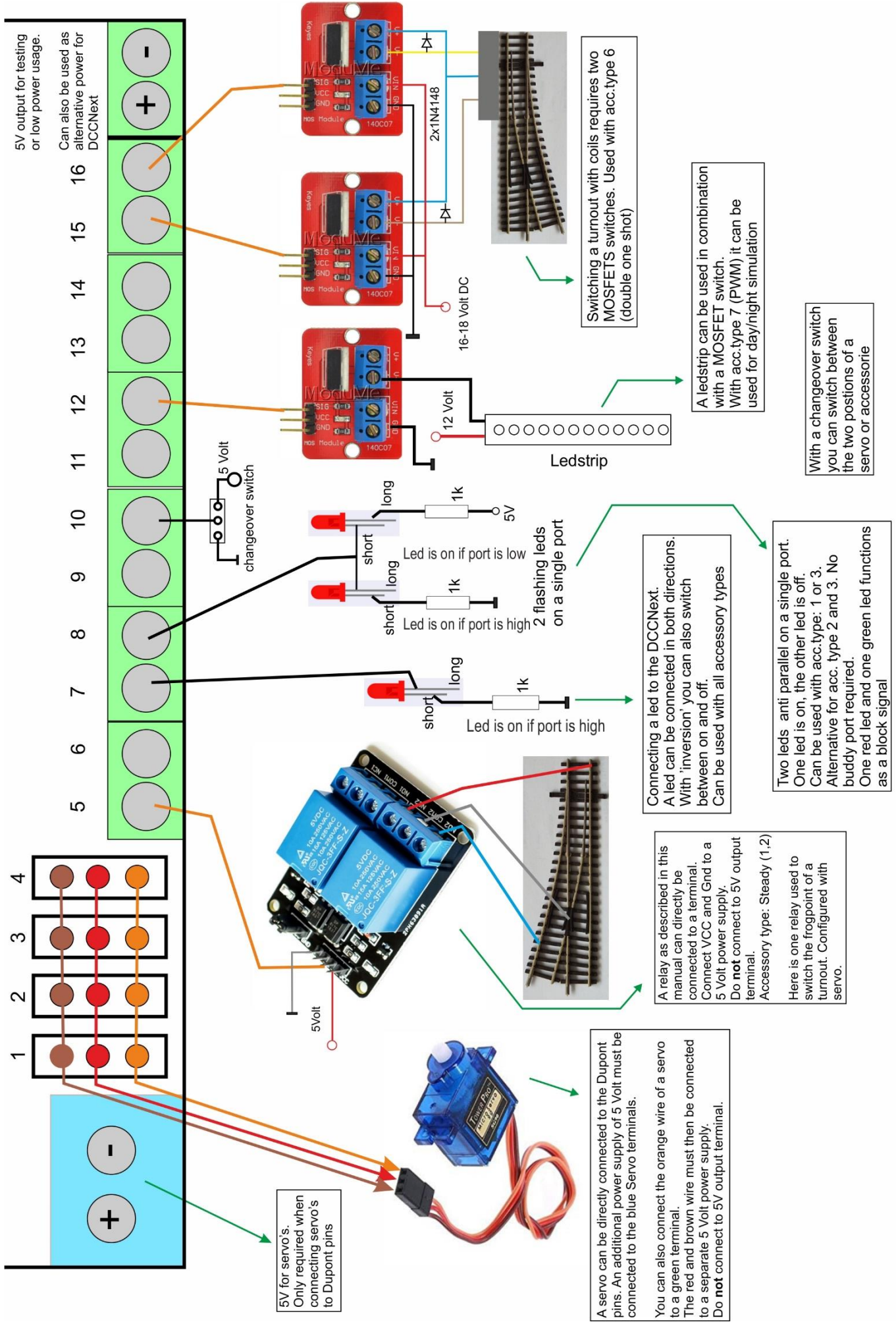
Of course, software is also required:

- 1) With the software installation, the MARDEC program is placed on your PC as a "pre-compiled" binary file: MARDEC.hex.
- 2) To get this back on the Arduino you need upload software. MARDEC is thus transferred from your PC to the Arduino via the USB cable. This upload is done by a script that is started with a shortcut on the desktop.
- 3) A separate "terminal emulator" is used for communication with MARDEC. (Putty.exe). This too is started via a shortcut. With this you can display the output of the program running on the Arduino and also input from your keyboard can be sent to MARDEC.
- 4) In addition, drivers may also be required. If you have already installed the Arduino IDE (Integrated Development Environment) you probably already have it. For Chinese clone Arduinos and the DCCNext you need special drivers, the so-called CH340 driver.

All these four components are put on your PC in a single installation.

See the installation manual for this.

Connecting the Arduino



Configuring Mardec

If all software is installed correctly and the Arduino is connected to your PC you can now configure Mardec.

To do so click on the shortcut *Configure Mardec*.

If you do this for the very first time with your first Arduino, the USB drivers for the Arduino will be activated followed by the upload of Mardec and the start of Putty. Putty is a terminal emulator for the communication between the Arduino and your PC. (see also installation manual).

Mardec uses two different modes. Configuration mode and operation mode. By default it starts up in configuration mode. But when restarted in operation mode it will start up in operation mode again.

Please Note:

- Every numeric input must be ended with <enter>.
- Only one-character commands don't require an <enter>
- For most inputs the current value does not change with only <enter>. The current value is shown between brackets.
- In configuration mode the status led on the DCC shield or the DCCNext is always on.
- In operation mode this led is off.
- Turn on the Numlock key on the numeric keypad.
- You may use lower and upper characters for commands.
- Use ONLY backspace key to correct an input.

The first time you start configuration mode or after a full reset (with R-Command) you have to enter some basic settings for your Mardec. The settings and their meaning can be found at the new general I-command

At every start Mardec shows an overview of the current settings.

The Commands

There are four types of commands:

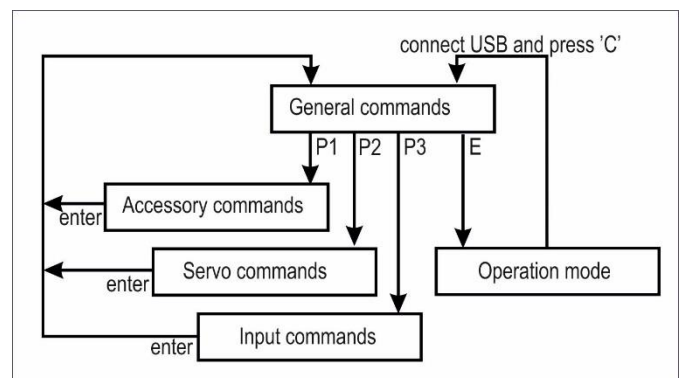
- **General** commands
- **Servo** bound commands
- **Accessory** bound commands
- **Input** bound commands

Therefore the configuration mode has four different states. By entering a '?' you will see a list list of the available commands in the current mode.

At start up the General command state is automatically activated.

Every command can be entered by a single character without <enter>.

The most important command is 'P' (for **P**ort). With the P-command you can define/modify an Arduino port.



General Commands

I-Command

With the I command, some initial settings are reset.

- **Administrative number** for this MARDEC. number identifies the decoder.
- Do you use a **Roco Multimaus**, z21 or Z21? this case the MARDEC will automatically make an address correction. (default = No)
- **NOTE: Switch off the Railcom/Rail communication function in the z21!**
- The standard **rotation speed** of the servos. This applies to newly added servos.
- Normally a servo is "disconnected" (= **detached**) from Arduino after reaching the end position. This is to prevent jittering. This allows the servo to rotate if an external force is applied. If that twist is a problem, you can also keep a servo "attached". Mardec will then immediately correct any deviation.
- Choose how you want Mardec **to start up**: As last mode, always in configuration mode or always in normal mode

```
mardec on port COM28
MARDEC, the Multifunctional ARduino dcc DECoder
UNO version 5.0

Mardec starting, please wait

Specify number for this MARDEC
Enter value from 1 to 100: 1
This is MARDEC #1

Do you use a ROCO MultiMaus or z/Z21 Y/N ? (def.: N)
Address offset: No

Default rotation speed for all servo's
Enter value from 5 to 100 (25):
Default speed set to 25 ms/degree

Detach servo at end of rotation Y/N ? (def.: Y)
Servo's are detached at end of rotation

Startup mode: 1=Last (def.),2=Configuration,3=Normal
Enter value from 1 to 3 (1):
Startup mode: Last
```

This
In
the

P-Command (Port)

After the P-command a port number will be asked. If port is not in use, you must first specify its DCC address the required function (servo, accessory or input) If the port is already in use then its current configuration is shown and the configuration state changes to the corresponding port type. You can use the port numbers 1 to 16 An accessory that uses PWM (mode 7 and 8) is only available on ports 1, 3, 4 and 9 of an UNO/DCCNext on ports 1 upto 8 on a MEGA Depending on your selection Mardec will be set to the corresponding command set.

```
Specify action (P/A/R/T/D/E/I/?): P
Specify port number
Enter value from 1 to 16: 1
Port is undefined. Specify DCC address

Set DCC address for port 1
Enter value from 1 to 2000: 236
DCC Address set to 236

Specify Accessory(1), Servo(2) or Input(3)
Enter value from 1 to 3: 1
```

the
and
and

A-Command (Address)

If a port is already configured you can also modify it by entering its DCC address. Mardec will find the correct port number

```
Specify action (P/A/R/T/D/E/I/?): A
Specify DCC address
Enter value from 1 to 2000: 236
DCC 236, Acc.type 1 (S. Steady), , Not Inv.

Specify action for Accessory on A236/P1
enter/A/N/I/D/T/M/H/L/?/F/R: █
```

port

D-Command (document)

The D-command shows an overview of the configuration of all ports. You may want to make a screen dump of it or copy paste the text to a file. So you always know what the configuration is of Mardec.

R-Command (Reset)

With the R command all settings can be deleted from the of the Arduino and confirmation is requested twice. After a reset, the screen will be cleared and the the permanent requested.

```
Specify action (P/A/R/T/D/E/I/?): R
Clear all settings? Y/N: Y
ABSOLUTELY sure? Y/N: Y
```

memory
settings

E-Command (Exit)

The E-command starts the run time mode.

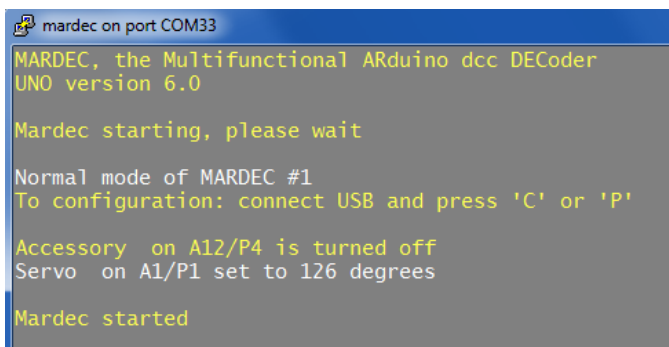
Now you can control the configured accessories with DCC signals from your command station.

The statusled is now off. It will flash when a configured DCC-address passes by.

Also an informative text is shown in the Putty screen.

However you may also close Putty and remove the USB

PLEASE NOTE: The E-command can also be entered while configuring a port. The settings made will be saved immediately and the operating state will be started.



```
mardec on port COM33
MARDEC, the Multifunctional ARduino dcc DECoder
UNO version 6.0
Mardec starting, please wait
Normal mode of MARDEC #1
To configuration: connect USB and press 'C' or 'P'
Accessory on A12/P4 is turned off
Servo on A1/P1 set to 126 degrees
Mardec started
```

cable.

Back to Configuration

To return to configuration mode, reconnect the USB cable. Then start the configuration program using the desktop shortcut. This will restart the Arduino as you can see by the 3 times short flashing of the status LED and then stays off. This will put MARDEC in the configuration state and the status LED will light up continuously again.

You can also press the 'P' key. Then the configuration mode will start again and the 'P' command will be issued and you will be asked for a port number.

This is the so-called **Quick-Config mode**

Important

After each received DCC command, the modified position of a servo or accessory is stored directly in the memory of the Arduino. When the Arduino is switched on, these stored states are read out again and the servos and accessories are reset as they were when the Arduino was shut down.

It is possible that a servo moves more or less during the power up.

Because Mardec, with some delay, puts the servos at the last set position, these start-up effects will no longer have any effect. A possible 'start-up rotation' is restored by Mardec. A good power supply remains a requirement as well as good quality servos (TG9d, HXT900, ES9051).

Optionally, a ferrite core near the servo can also help.

A resistor of 4k7 between signal and 5V can also help.

Quick Config Mode

In configuratie modus:

Enter command (P/A/D/E/I/R/?): **p**

Select port number.

Enter value from 1 to 16: 6

DCC 55, Acc.type 3 (S. Flashing), , Not Inv., Time(ms) 200/300

Enter command for Accessory on A55/P6

M/I/R/T/A/D/N/E/enter/?: **m** **Modify**

Select mode for this accessory

0=Help. Enter value from 0 to 9 (3):

Mode set to Single Flashing

Enter 'On' time in millisec.

Enter value from 5 to 30000 (200):

'On' time set to 200 msec.

Enter 'Off' time in millisec.

Enter value from 5 to 30000 (300): **400** **Off time changed**

'Off' time set to 400 msec.

Enter command for Accessory on A55/P6

M/I/R/T/A/D/N/E/enter/?: **E** **Direct E-command**

Port settings are saved

Normal mode starts

MARDEC, the Multifunctional ARduino dcc DECoder

UNO version 6.0

Mardec starting, please wait

Normal mode of MARDEC #1

To configuration: connect USB and press 'C' or 'P'

Accessory on A12/P4 is turned off

Accessory on A55/P6 stopped flashing

Servo on A1/P1 set to 126 degrees

Mardec started

P enter P

configuration mode starts

MARDEC, the Multifunctional ARduino dcc DECoder

UNO version 6.0

Mardec starting, please wait

Configuration mode of MARDEC #1

....

....

Mardec started

Select port number. **Automatic P-command**

Enter value from 1 to 16: 6

DCC 55, Acc.type 3 (S. Flashing), , Not Inv., Time(ms) 200/**400**

Enter command for Accessory on A55/P6

M/I/R/T/A/D/N/E/enter/?:

With the Quick Config mode it is possible to quickly make a change for a port from the operating state. Entering 'P' starts config mode and ask for a port number. After the change, the port does not have to be saved first by means of <enter>, but the E command can be given directly.

Input commands

With a port that is configured as an input port you can simulate a DCC signal for a servo or accessory that has the **same address** as the input port.

Example:

On port 12 is a single flashing led configured with DCC address 34.

On port 5 is an input configured with also DCC address 34.

By default an input port is kept high by the Arduino. If it is made 'low' by e.g. mechanical switch or current sensing print. Mardec interprets it as a DCC signal for address 34 and the flashing led on port 12 stops (or starts) flashing. If there is also a servo on address 34 it starts rotating.

An input can be activated in three ways.

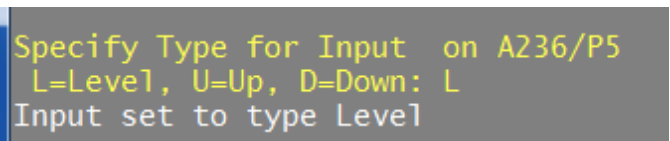
- 1) By lowering the input. Nothing happens when the entrance becomes high again. (type **Down**)
- 2) By making the entrance high. Nothing happens when the entrance becomes low again. (type **Up**)
- 3) By making the input low or high. The accessory or servo "follows" the input signal as if it is controlled by DCC. (type **Both**)

With this option, for example, a turnout can also be converted by means of a switch.

With options 1 and 2, an input acts as a momentary switch.

With option 3, an input acts as a changeover switch. Just like command.

The activation method is requested when configuring a port.



a DCC

An input port can be controlled in various ways. For example, by a reed switch that switches to ground or a current detection signal that becomes low when a train "passes". ([Okkie](#))

An infrared light barrier or just a changeover switch can also be used.

The attributes at address 34 can of course also be controlled from a real DCC signal. An important difference, however, is that nothing happens with an input port if, when using options 1 or 2, it becomes low or high again, while with a DCC signal it does.

With option 3 (Level), an input works the same as the DCC signal.

PAY ATTENTION:

Remember that if you switch on an accessory from the command station/computer and then switch it off again via an input port, the command station/computer still "thinks" that the accessory is switched on.

Turning it off again using DCC therefore no longer has any effect.

2- and 3-Commands (second and third address)

Also a second and third address can be connected to an input port. Therefore you can use the 2 and 3 command. By activating the input port the accessories on these second and third address will also be activated.

For the second and third address you can also specify a delay time up to 25 seconds in steps of

0.1 seconds. The accessory on the 2nd and 3rd address will then activated after the delay time has passed. Both delay times are relative to the moment of activating the input port.

In the figure on the right first the accessory on address 20 ('own' address) will be activated, when activated by port 5. There is no delay possible.

After 5 sec. the accessory on address 25 (2nd address) is activated and after 7 sec. the accessory address 26 (3rd address). All addresses must, of course, be configured on the same Arduino.

If you only want 'delayed' actions you can configure a dummy address as 'own' address of the input port. Also you can specify the address of another input port as a 2nd or 3rd address! This allows you to start a whole range of actions.

```

Specify port number
Enter value from 1 to 16: 5
Port is undefined. Specify DCC address

Set DCC address for port 5
Enter value from 1 to 2000: 236
Address in use for servo/accessory on A236/P1
Is that OK ? (Y/N, N=default): Y

DCC Address set to 236

Specify Accessory(1), Servo(2) or Input(3)
Enter value from 1 to 3: 3
Port 5 set as Input port.

Specify Type for Input on A236/P5
L=Level, U=Up, D=Down: L
Input set to type Level

Specify action for Input on A236/P5
A/N/?/D/T/2/3/enter: 2
Specify Second control address.
Enter value from 1 to 2000: 120
Specify delay time in 0.1 seconds.
Enter value from 0 to 250: 20
Address 120 set as Second control address with a delay of 2.0 sec.

Specify action for Input on A236/P5
A/N/?/D/T/2/3/enter: 3
Specify Third control address.
Enter value from 1 to 2000: 45
Specify delay time in 0.1 seconds.
Enter value from 0 to 250: 50
Address 45 set as Third control address with a delay of 5.0 sec.

Specify action for Input on A236/P5
A/N/?/D/T/2/3/enter:

Port settings are saved!

Specify action (P/A/R/T/D/E/I/?): D

Settings of MARDEC #1

Default servo rotation speed: 25 ms/degree
Address offset: No
Servo's are detached at end of rotation
Startup mode: Last

Port 1: DCC 236, Acc.type 1 (S. Steady), , Not Inv.
Port 2: DCC 236, Servo , Angles 75/105, Not Inv., Speed 25, Frog port no
Port 3: DCC 120, Acc.type 3 (S. Flashing), , Not Inv., Time(ms) 300/400
Port 4: DCC 45, Servo , Angles 75/105, Not Inv., Speed 25, Frog port 16
Port 5: DCC 236, Input , Type: L, Second addr/delay: 120/2.0 sec, Third addr/delay: 45/5.0 sec

```

In this example, the following happens if port 5 (= input) changes level:

- The accessory on port 1 turns on or off because port 1 has the same address as port 5.
- The servo on port 2 starts running because port 2 has the same address as port 5.
- After 2 seconds, the accessory on port 3 turns on or off because port 3 has the same address as the second address of port 5.
- After 5 seconds the servo on port 4 starts running because port 4 has the same address as the third address of port 5.

If DCC address 236 is sent:

- The accessory on port 1 switches on or off.
- The servo on port 2 starts to turn.

If DCC address 120 is sent: the accessory on port 3 turns on or off.

If DCC address 45 is sent: the servo on port 4 starts running.

A-Command (Address)

The DCC address of the input is set with the

A-command. You will receive a warning if the address has already been assigned to another servo, accessory or input. By accepting that you can control multiple servos/accessories with one address.

```
Select port number.
Enter value from 1 to 16: 1
DCC 1, Input , Trigger: B, Second addr/delay: none, Third addr/delay: none

Enter command for Input on A1/P1
T/2/3/enter/A/N/D/? : a

Set DCC address for port 1
Enter value from 1 to 2000: 5
DCC Address set to 5
```

N-Command (Note)

With the N-command you can assign an administrative code to the input port. It has no technical meaning whatsoever and entry is also not required. (4 characters)

T-command (Trigger)

With the T command you can change the trigger type of the input.

```
Enter command for Input on A5/P1
T/2/3/enter/A/N/D/? : T

Select trigger moment Up(U), Down(D) or Both(B): D
Trigger set to type Down-Pulse
```

R-Command (Reset)

With the R-command you can reset the port. After that the port can be reconfigured again.

D-command (Document)

The D command shows an overview of all settings.

<enter or X>-Command

Entering <enter> or X saves the settings of the port and returns to the general command s A different port can then be selected.

```
Enter command (P/A/D/E/I/R/?): p

Select port number.
Enter value from 1 to 16: 1
DCC 1, Acc.type 1 (S. Steady), , Not Inv.

Enter command for Accessory on A1/P1
M/I/R/T/A/D/N/enter/? : r

Do you want to reset port 1? Y/N [N]: y
Port 1 is now reset

Port settings are saved!

Enter command (P/A/D/E/I/R/?): █
```

?-Command (Help)

This shows the available input commands s

I-Command (Invert)

With the I command the input is inverted. A high signal on the input is then treated as a low signal and vice versa.

E-Command (Exit)

The E command is used to save the portchanges directly and to start the operating state. It is then not necessary to end the port configuration with <enter> or X first.

To test

There is no separate test function for input ports. However, you can test them by going to operating mode with the Exit command. By connecting the input port to a gnd port with a cable, you can check whether the linked servo/accessory responds correctly.

Servo commands

A dialog for configuring a servo looks as follows:

A new servo is set to angles of 75(low) and 105(high) degrees. An existing servo is set to its already configured angles. In both cases the servo is set to the low position.

- (minus) Command

The '-' command lowers the angle with one degree. The high angle must be at least 5 degrees higher than the low angle.

+ (plus) Command

The '+' command raises the angle with one degree. The low angle must at least 5 degrees lower than the high angle

9-Command

With the 9-command you can set a servo to 90 degrees. This is useful for placing the lever of the servo in the middle. Do this before mounting the servo under the track.

C-Command (Change)

The C-command lets you toggle between both angles. So you can set both angles individually by using the - and + key.

I-Command (Invert)

It depends on how the servo is mounted under the track whether the turnout goes straight at the low angle or high angle of the servo.

With the I-command you can invert the rotation direction of the servo.

T-Command (Test)

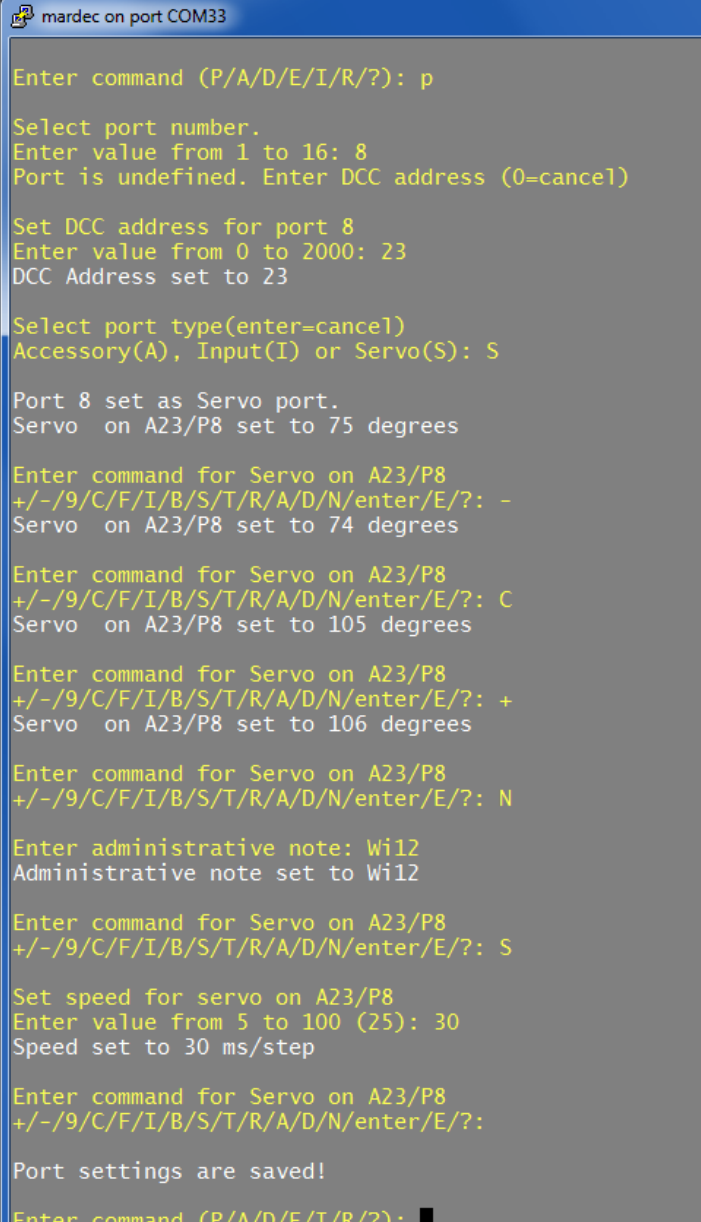
The T command is the same as the C command. In addition, the simulated DCC value is shown. However, the change is done by directly setting the end angle.

S-Command (Speed)

With the S-command you can set the individual speed of a servo between 5 (fast) and 100 (slow) ms. per degree. The default is 25 ms, which can be set with the generic S-command.

?-Command (Help)

This shows all available servo commands



```
mardec on port COM33
Enter command (P/A/D/E/I/R/?): p
Select port number.
Enter value from 1 to 16: 8
Port is undefined. Enter DCC address (0=cancel)

Set DCC address for port 8
Enter value from 0 to 2000: 23
DCC Address set to 23

Select port type(enter=cancel)
Accessory(A), Input(I) or Servo(S): S

Port 8 set as Servo port.
Servo on A23/P8 set to 75 degrees

Enter command for Servo on A23/P8
+/-/9/C/F/I/B/S/T/R/A/D/N/enter/E/? : -
Servo on A23/P8 set to 74 degrees

Enter command for Servo on A23/P8
+/-/9/C/F/I/B/S/T/R/A/D/N/enter/E/? : C
Servo on A23/P8 set to 105 degrees

Enter command for Servo on A23/P8
+/-/9/C/F/I/B/S/T/R/A/D/N/enter/E/? : +
Servo on A23/P8 set to 106 degrees

Enter command for Servo on A23/P8
+/-/9/C/F/I/B/S/T/R/A/D/N/enter/E/? : N

Enter administrative note: Wi12
Administrative note set to Wi12

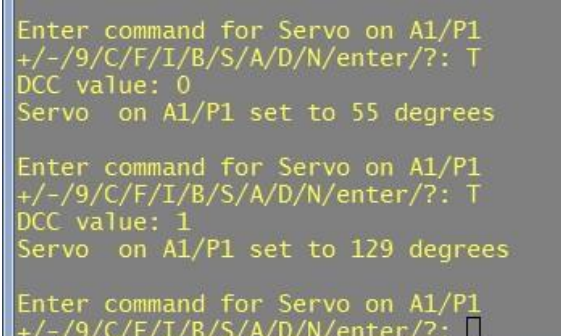
Enter command for Servo on A23/P8
+/-/9/C/F/I/B/S/T/R/A/D/N/enter/E/? : S

Set speed for servo on A23/P8
Enter value from 5 to 100 (25): 30
Speed set to 30 ms/step

Enter command for Servo on A23/P8
+/-/9/C/F/I/B/S/T/R/A/D/N/enter/E/? :

Port settings are saved!

Enter command (P/A/D/E/I/R/?): █
```



```
Enter command for Servo on A1/P1
+/-/9/C/F/I/B/S/A/D/N/enter/? : T
DCC value: 0
Servo on A1/P1 set to 55 degrees

Enter command for Servo on A1/P1
+/-/9/C/F/I/B/S/A/D/N/enter/? : T
DCC value: 1
Servo on A1/P1 set to 129 degrees

Enter command for Servo on A1/P1
+/-/9/C/F/I/B/S/A/D/N/enter/? : █
```


N-Command (Note)

I'm sure you've given the turnouts and accessories a code in your track design. This administrative code can also be assigned to the port using the N command.

It has no technical meaning and entry is not required. You can enter 4 characters.

B-command(Bouncing)

With the B-command you can let the servo arm bounce a couple of times (0-4) at the end of the rotation.

The angle with which the arm turns back can be adjusted from 1 to 9 degrees.

The angles are entered as a number of maximum 4 digits.

Example 1: A value of 7531 means that the servo first goes back 7 degrees and then back to the end position.

Then the servo goes back 5 degrees and back to the end position.

Then the servo goes back 3 degrees and back to the end position. Finally the servo goes 1 degree back and back to the end position.

Example 2: With a value of 42, the servo will first thunder back 4 degrees and then 2 degrees before it comes to rest.

The bounce angles are separately adjustable for both end positions!

```
Enter command for Servo on A1/P1
+/-/9/C/F/I/B/S/A/D/N/enter/?: B
Set bounce angles for Low angle
Enter value from 0 to 9999: 7531
Set bounce angles for High angle
Enter value from 0 to 9999: 42
Bounce angles set to (L-H) 7531-42
```

A-Command (Address)

The A command is used to set the DCC address of the servo. You will receive a warning if the address has already been assigned to another servo, accessory or input. By accepting that you can control multiple servo's/accessories at the same time with one address.

D-command (Document)

The D command shows an overview of all settings.

R-Command (Reset)

With the R-command you can reset the port.

After that the port can be reconfigured again.

```
Enter command (P/A/D/E/I/R/?): p
Select port number.
Enter value from 1 to 16: 1
DCC 1, Acc.type 1 (S. Steady), , Not Inv.
Enter command for Accessory on A1/P1
M/I/R/T/A/D/N/enter/?: r
Do you want to reset port 1? Y/N [N]: y
Port 1 is now reset
Port settings are saved!
Enter command (P/A/D/E/I/R/?): █
```

<enter or X>-Command

Entering <enter> or X saves the servo settings and returns them to the general command's. A different port can then be selected.

E-Command (Exit)

The E command is used to save the port changes directly and to start the operating state.

It is then not necessary to end the port configuration with <enter> or X first.

F-Command (Frog point)

With the F-command you can specify a port number to which you can connect a relay for frog point polarization.

If you want one relay, enter a '1', or a '2' for two relays.

(see below)

MARDEC will now ask to which port you want to connect the first (and possibly second) relay.

If a frog point is already assigned, Mardec asks whether you want to remove the frog point or toggle the relay inversion.

```

Enter command (P/A/D/E/I/R/?): p
Select port number.
Enter value from 1 to 16: 4
DCC 43, Servo , Angles 75/105, Not Inv., Speed 25, Frog port no, Bounce (L/H): 0/0
Angle set to LOW: 75

Enter command for Servo on A43/P4
+/-/9/C/F/I/B/S/A/D/N/enter/? : F
How many relays? Enter value from 0 to 2: 2
Set Frog port. Enter value from 1 to 16: 8
Port 8 set as first frog port
Set second Frog port. Enter value from 1 to 16: 9
Port 9 set as second frog port

Enter command for Servo on A43/P4
+/-/9/C/F/I/B/S/A/D/N/enter/? : 
    
```

When changing the turnout, the relay will also be converted exactly halfway through the rotation of the servo. When twisting to the largest angle, the relay will be energized and when twisting to the smallest angle, the relay will fall off. Because the servo itself also has an inversion option, you can adjust the position of the turnout and the point polarisation to each other entirely using software.

If there is already a frog port assigned, MARDEC asks if you want to remove it (R) or reverse the inversion of the relay '1' for first relay, '2' for second relay).

```

Specify action for Servo on A32/P4
A/N/9/+/-/C/F/D/T/S/I/?/enter: F

Remove frogpoint or Toggle Port relay inversion? R/1/2: 1
First Relay inverted
    
```

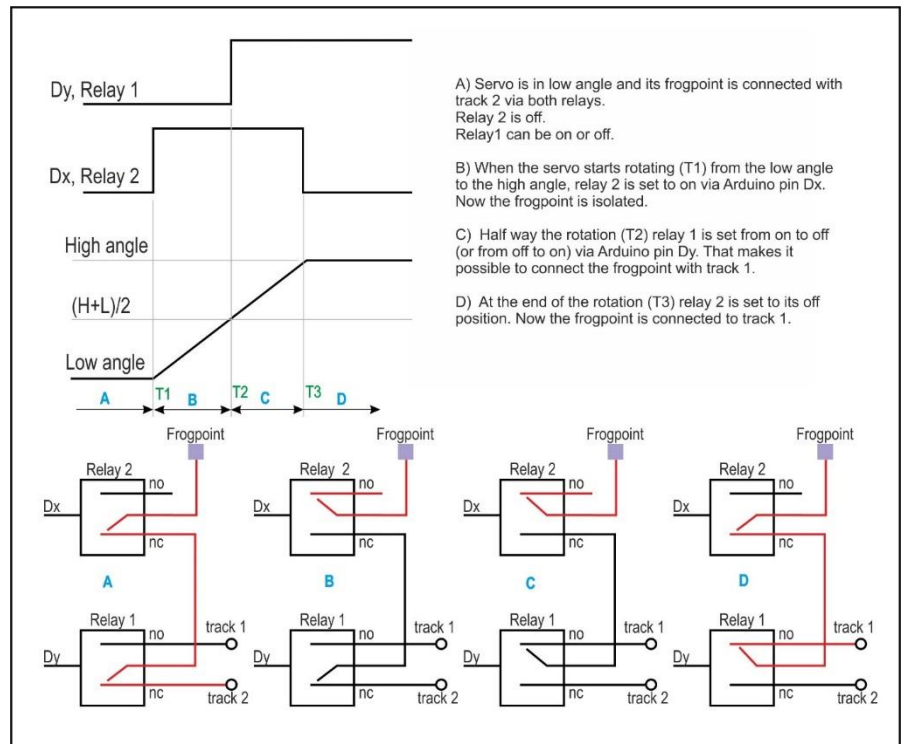
You can also choose for two relays. The second relay is then used to completely isolate the frog point during the entire rotation. For two relays you enter a '2'.

Because some relays are low active (are powered by 0 volts instead of 5 volts) you can also set for the second relay inversion. The relay module as described on page 7 is such a low active relay see also page 27. Initially the relays are set to no inversion. By entering the F-command again, you can invert the relays..

Some relays are 'active-low' (activated with 0 Volt instead of 5 V). By toggling the inversion for relay 2 you can be sure that the relay is only activated when the servo is rotating.

TIP: You can of course also connect something else than a relay to the frog port; e.g. operate a switch signal or an indication LED that shows the position of the switch.

You can also use the relay to switch something different then the frog point. Actually, the frog port is simply a port that becomes high or low at the same time as the switch is switched.



Accessory Functions(=mode)

The data of the DCC signal has only two values (0/1, on/off, low/high, straight/rounded, red/green)
MARDEC realizes several functions with this data.

All the 'double' functions specified below(2, 4 and 6) requires the use of a second port. This is the so called 'buddy' port.
You can decide for yourself, MARDEC will already ask for it: *Set Buddy port. Enter value from 1 to 16:*

For a new port MARDEC asks the required function (mode).
For an existing port the current configuration is shown.
The following modes are available:

Enter command (P/A/D/E/I/R/?): **P**
Select port number.
Enter value from 1 to 16: **1**
Port is undefined. First enter DCC address

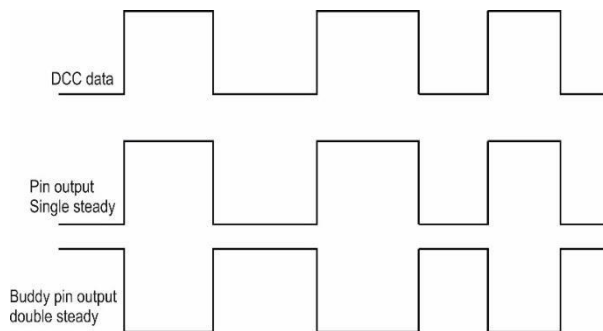
Set DCC address for port 1
Enter value from 0 to 2000: **1**
DCC Address set to 1

Select port type: Accessory(A), Servo(S) or Input(I): **A**
Port 1 set as Accessory port.

Select mode for this accessory
0=Help. Enter value from 0 to 9 (9): **0**
1 Single steady, 2 Double steady
3 Single flashing, 4 Double flashing
5 Single One shot, 6 Double One shot
7 Analog PWM, 8 Flickering
9 Random On/Off

Mode 1, Single Steady

Enter value from 0 to 9: **1**
Mode set to Single Steady



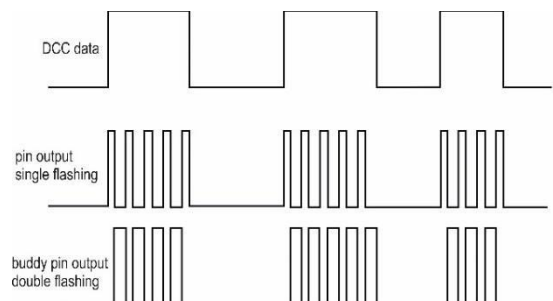
With **mode 1** the port 'follows' the DCC data

Enter command for Accessory on A1/P1
M/I/R/T/A/D/N/enter/? : **M**

Mode 2, Double Steady

Select mode for this accessory
0=Help. Enter value from 0 to 9 (1): **2**
Set Buddy port. Enter value from 1 to 16: **2**
Port 2 set as buddy port
Mode set to Double Steady

With **mode 2** the port 'follows' the DCC data
And a second port (buddy port) has the inverted value



Enter command for Accessory on A1/P1
M/I/R/T/A/D/N/enter/? : **M**

Mode 3, Single Flashing

Select mode for this accessory
0=Help. Enter value from 0 to 9 (2): **3**
Mode set to Single Flashing
Enter 'On' time in milliseconds Enter value from 5 to 30000: **200**
'On' time set to 200 msec.
Enter 'Off' time in milliseconds Enter value from 5 to 30000: **400**
'Off' time set to 400 msec.

In **mode 3**, the port is alternately high and low as long as the DCC data is high.
The on and off times are separately adjustable between 5 and 30,000 ms. (=30 sec.)

Enter command for Accessory on A1/P1

M/I/R/T/A/D/N/enter/?: **M**

Mode 4, Double Flashing

Select mode for this accessory

0=Help. Enter value from 0 to 9 (3): **4**

Set Buddy port. Enter value from 1 to 16: **2**

Port 2 set as buddy port

Mode set to Double Flashing

Enter 'On' time in milliseconds

Enter value from 5 to 30000: **500**

'On' time set to 500 msec.

Enter 'Off' time in milliseconds

Enter value from 5 to 30000: **100**

'Off' time set to 100 msec.

Enter command for Accessory on A1/P1

M/I/R/T/A/D/N/enter/?: **M**

Mode 5, Single One Shot

Select mode for this accessory

0=Help. Enter value from 0 to 9 (4): **5**

Mode set to Single One shot

Enter 'On' time in milliseconds

Enter value from 5 to 30000: **500**

'On' time set to 500 msec.

Select trigger moment Up(U), Down(D) or Both(B): **D**

Trigger set to type Down-Pulse

Enter command for Accessory on A1/P1

M/I/R/T/A/D/N/enter/?: **M**

Mode 6, Double One Shot

Select mode for this accessory

0=Help. Enter value from 0 to 9 (5): **6**

Set Buddy port. Enter value from 1 to 16: **2**

Port 2 set as buddy port

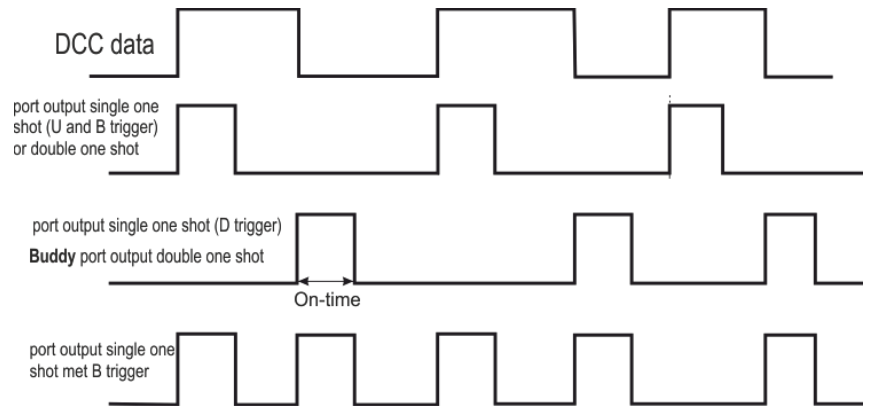
Mode set to Double One shot

Enter 'On' time in milliseconds

Enter value from 5 to 30000: **1000**

'On' time set to 1000 msec.

Mode 4 is the same as single flashing but a buddy port gets the "reversed" value.



With mode 5, the port becomes high (5-30.000 ms) for a certain period of time when the DCC data becomes high (U or B trigger) or low (D or B trigger).

If the address of a single one shot is equal to that of another servo or accessory, at the end of the pulse, it will be 'converted' as if a DCC signal had been sent.

Example: At address 12 both a single one shot and a double flashing are configured. If MARDEC now 'sees' address 12, both the one shot and the double flashing will be started. At the end of the pulse, the double flashing will be switched off again. This allows an accessory to be activated for a short time (max. 30 sec.).

The double one shot (mode 6) is the same as the single one shot. However, the double one shot is only started when the DCC data becomes high.

A buddy port goes high when the DCC data becomes low again.

With this function you can also control alternating coils.

Alternating coils require at least 12 volts and draw a relatively large current. Therefore you cannot connect them directly to an Arduino port. This requires an extra amplifier stage.

See the example in the schematic (port 16).

For a turnout, you need two amplifier stages. One for each coil.

The MOSFET modules mentioned on page 10 are suitable for this.

Enter command for Accessory on A1/P11
M/I/R/T/A/D/N/enter/?: **M**

Mode 7, Analog PWM

Select mode for this accessory

0=Help. Enter value from 0 to 9 (6): **7**

Mode set to Analog PWM value

Set Minimum PWM value for port A1/P1

Enter value from 0 to 255: **5**

Minimum PWM value set to 5

Set Maximum PWM value for port A1/P1

Enter value from 5 to 255 (255): **200**

Maximum PWM value set to 200

Set Rise time (sec) for PWM port

Enter value from 0 to 1000: **60**

Rise time set to 60 sec.

Set Fall time (sec) for PWM port

Enter value from 0 to 1000: **30**

Fall time set to 30 sec.

Enter command for Accessory on A1/P1

M/I/R/T/A/D/N/enter/?: **M**

Mode 8, Flickering (PWM)

Select mode for this accessory

0=Help. Enter value from 0 to 9 (7): **8**

Mode set to Flickering(PWM)

Set Minimum PWM value for port on A1/P1

Enter value from 0 to 255: **2**

Minimum PWM value set to 2

Set Maximum PWM value for port on A1/P1

Enter value from 2 to 255 (255): **200**

Maximum PWM value set to 200

Enter command for Accessory on A1/P1

M/I/R/T/A/D/N/enter/?: **M**

Mode 9, Random aan/uit

Select mode for this accessory

0=Help. Enter value from 0 to 9 (1): **9**

Mode set to Random

Fixed on/off time? 1=No, 2=ON, 3=OFF

Enter value from 1 to 3: **2**

Fixed ON time(x0,02 sec) ?

Enter value from 1 to 30000: **1000**

Fixed ON time set to 20.00 sec.

Set Minimum random time (x0.02 sec)

Enter value from 1 to 30000: **500**

Minimum random time set to 10.00 sec.

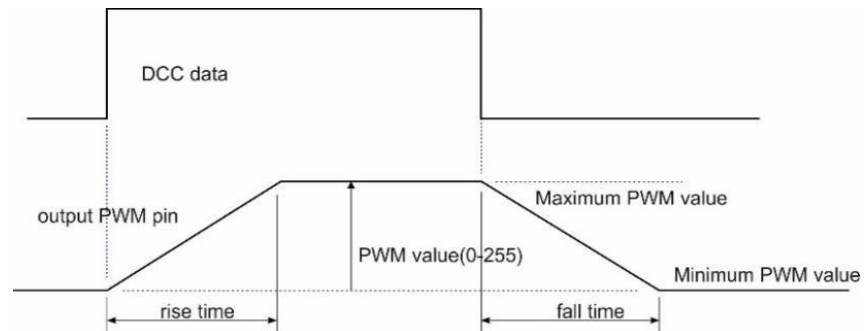
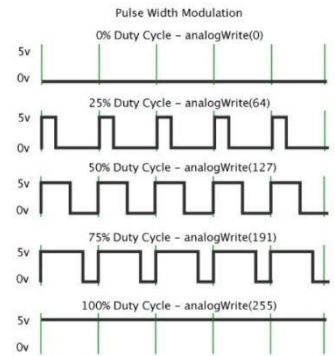
Set Maximum random time (x0.02 sec)

Enter value from 500 to 30000: **4000**

Maximum random time set to 80.00 sec.

Mode 7: On a number of ports (1,3,4 and 9; on Mega ports 3 to 10) an Arduino can set a block voltage of which the pulse width (duty cycle) can vary from 0 to 100%.

This is realized by 'writing' a value from 0 to 255 to this port. By e.g. connecting a led and vary the PWM value, this LED can be dimmed. This can be used for e.g. variable track lighting such as day/night cycles.



The following can be set for this function:

- the maximum PWM value (min. value - 255)
- the minimum PWM value (0 - max value)
- the rise time (0-1000 sec)
- the fall time (0-1000 sec)

With three of these accessories you can, for example, control an RGB LED strip. Also a led strip needs an amplifier stage in the form of a MOSFET transistor.

With **mode 8** you can flicker a connected led with an adjustable minimum and maximum brightness.

Every 20 msec a random value is set for the brightness.

By flickering three suitable LEDs in different colors, you can simulate fire.

It can also be used for e.g. welding light simulation.

PWM is also used and can only be used on ports 1, 3, 4 and 9. (On Mega ports 3 to 10)

With **mode 9**, a connected accessory (e.g. an LED) will switch on and off continuously as long as the DCC data is high. For each on and off period a time is randomly determined.

The minimum and maximum time can be set between 20 msec. and 600 seconds in 20 msec. increments.

A fixed on/off time or completely random can be chosen.

If a fixed on time is set, the off time is random. If a fixed off time is set, the on-time is random.

The random function can be used to turn on and off e.g. interior lighting of houses or street lighting.

With very short times the flickering of a lamp can be simulated.

Accessory Commands

? Command

Shows available options

M-Command (Mode)

With the M command you can choose another function for the selected port. It will ask for the specific settings of the new function.

You can also use the same function to change the port settings or, in the case of a double function, to change the buddy port.

```
Enter command for Accessory on A22/P2
M/I/R/T/A/D/N/enter/? : ?

I Invert port
T Test port
M Modify mode
A DCC-Address
D Display all
N admin. Note
R Reset port
Enter/X Update Port
```

R-Command (Reset)

The R command allows you to reset the port. After that the port can be reconfigured.

D command (Document)

The D command shows an overview of all settings

A-Command (Address)

With the A command you can change the DCC address of the port.

```
Enter command (P/A/D/E/I/R/?): p
Select port number.
Enter value from 1 to 16: 1
DCC 1, Acc.type 1 (S. Steady), , Not Inv.

Enter command for Accessory on A1/P1
M/I/R/T/A/D/N/enter/? : r

Do you want to reset port 1? Y/N [N]: y
Port 1 is now reset

Port settings are saved!

Enter command (P/A/D/E/I/R/?): █
```

N-Command (Note)

The N command allows you to assign an administrative code to the gate. It has no technical meaning and entry is not mandatory. (4 characters)

T command (Test)

The T command is identical for all accessory functions.

By repeatedly entering 'T', the DCC signal is simulated and alternately set to 0 and 1.

This allows the connected accessory to be tested for its intended function.

Entering <enter> ends the test and the accessory menu is displayed again.

The E-command can also be entered to go directly to the operating state.

<enter or X>-Command

Entering <enter> or X saves the port settings and returns to the General commands Another port can then be selected.

E-Command (Exit)

The E command is used to save the gate changes directly and to start the operating state.

It is then not necessary to end the port configuration with <enter> or X first.

I-Command (Inversion)

With the I command you can reverse the output of a port if you wish.

The following table shows what inversion means per mode.

Mode 1		Single Steady	
	DCC	Port	
Not Inv	L	L	
	H	H	
Inv.	L	H	
	H	L	

Mode 2		Double Steady		
	DCC	Port	Buddy	
Not Inv	L	L	H	
	H	H	L	
Inv.	L	H	L	
	H	L	H	

Mode 3		Single Flashing		
	DCC	Port on	Port off	
Not Inv	L	L	L	
	H	H: on time	L: off time	
Inv.	L	L: off time	H: on time	
	H	H	H	

Mode 4		Double Flashing			
	DCC	Port on	Port off	Buddy on	Buddy off
Not Inv	L	L	L	L	L
	H	H: on time	L: off time	L: off time	H: on time
Inv.	L	L: off time	H: on time	H: on time	L: off time
	H	H	H	H	H

Mode 5		Single One Shot			
	DCC	Port (U)	Port (D)	Port (B)	
Not Inv	L > H	H pulse	L	H pulse	
	H > L	L	H pulse	H pulse	
Inv.	L > H	L pulse	H pulse	L pulse	
	H > L	H	L pulse	L pulse	

Mode 6		Double One Shot		
	DCC	Port	Buddy	
Not Inv	L > H	H pulse	L	
	H > L	L	H pulse	
Inv.	L > H	L pulse	H pulse	
	H > L	H pulse	L pulse	

Mode 7		Analog PWM	
	DCC	Port	
Not Inv	L > H	rising	
	H > L	falling	
Inv.	L > H	falling	
	H > L	rising	

Mode 8		Flickering PWM	
	DCC	Port	
Not Inv	L	L	
	H	flickering	
Inv.	L	flickering	
	H	H	

Mode 9		Random on/off			
	DCC	Fixed On	Fixed off	No Fixed	
Not Inv	L	L	L	L	
	H	Fix: H/Ra: L	Fix: L/Ra: H	Ra. H-L	
Inv.	L	H	H	H	
	H	FFix: L/Ra:	Fix: H/Ra: L	Ra H-L	

Servo		
	DCC	Angle
Not Inv	L	Low
	H	High
Inv.	L	High
	H	Low

Frog port		
	Angle	1-Relay
Not Inv	Low	L
	High	H
Inv.	Low	H
	High	L

Frog port		
	Servo	2-Relay
Not Inv	Rotates	H
	Stopped	L
Inv.	Rotates	L
	Stopped	H

H = Hoog, High, Hoch, Haute = 5 Volt

L = Laag, Low, Niedrig, Bas = 0

Miscellaneous

Boot up

When starting up the Arduino, the status LED will flash briefly three times. You can see that the Arduino is started.

Mount servos

Use the following procedure to set up a servo.

- Connect the servo to the Arduino and set the angle to 90 degrees with the 9-command.
- Now mount the servo under the track. Do this so that the switch points are about halfway along the tracks.
- Go with the C-command to one of the corners.
- Use the + and - commands to correct the angle.
- Go with the C-command to the other corner.
- Also adjust this.
- With point polarization: check the connection between the point piece and the rails. If wrong then invert adapter relay.
- Check whether the switch position corresponds to the symbol on the control panel / program. If wrong then invert servo.
- Exit with <enter> to save the settings.

Shut down

The control panel (Putty) can be closed at any time. No separate command is available for this.

So use Alt-F4 or click on the familiar cross in the top right-hand corner of the window.

Make sure that you are in the general command state, so that the last changes for a port are saved.

Logging

The entire configuration dialog is logged in the file:

My Documents / Mardec / MARDEC_<date>_<time>.log.

In addition, <date>_<time> is the time of closing.

Document

When you have finished configuring it is wise to record the settings in a document. For this purpose there is the shortcut Document your MARDEC in the MARDEC folder of the Windows start menu.

This opens an MS Word document that you can fill in. (So requires MS Word on your PC).

Arduino Nano, Pro Mini, Mega2560

You can use a Pro Mini in the same way as the UNO. Of course you can not use the DCC shield.

To upload Mardec to a Nano you must use the Arduino IDE. However, the default boot loader is too large, which means there is too little memory available for Mardec. However, you can replace the bootloader.

See the document '**fixing the boot loader**' at <https://www.arcomora.com/download/>

Then you can use the IDE to upload the .INO file.

To upload Mardec to a Mega2560 you must use the tool 'Upload' in the Windows start menu.

You can use the DCC shield for the Mega.

On the MEGA, ports 11 to 16 of the shield are connected to the gates 54 to 59 of the MEGA.

However, Mardec 5.0 will control the ports 54 to 59 on a MEGA as if it were the ports 11 to 16

So you can simply use the screw terminals from 11 to 16 as if the shield was on a UNO.

Configure subsequent decoders

After the software has been installed, MARDEC will be automatically loaded onto the Arduino when starting up 'Configure MARDEC' for the first time.

There is another method for the second and subsequent decoders.

To do this, start the shortcut 'Upload'.

You **must** use this if you have a MEGA2560. With the standard installation, an upload is always done to an UNO.

You can find this tool and its manual in the Windows start menu in the folder Arcomora.

Adjust com port

The Windows Com port sometimes changes when reconnecting an Arduino.

With the Change COM port tool you can easily adjust the Com port.

You can also find this tool in the Windows start menu

Configuration screen (Putty)

You can adjust the display of the control panel as follows.

1. Click the Configure Putty shortcut.

You can find these in the Windows start menu in the folder Arcomora

The adjacent screen appears:

2. Click on MARDEC and then on Load.

3. At Session Logging you can adjust the log options.

Note: If you change the filename, the configuration sessions can not be saved!

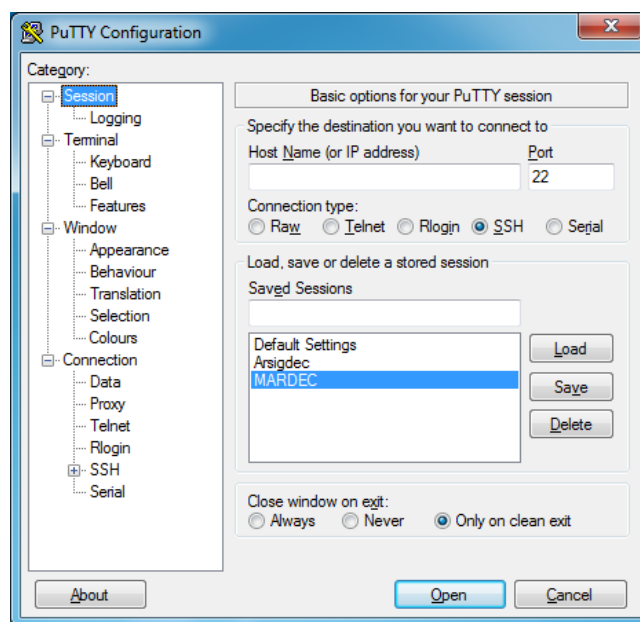
4. At Window Appearance / Behavior / Colors you can also adjust the appearance.

5. If necessary, you can also change the COM port here.

6. Do not change anything to all other settings!

7. Select Session and click Save to save the settings again.

8. Click Open to open the control panel again.



You can also easily change the COM port of an Arduino with the Change COM port program.

You can find this in the Windows start menu, in the folder Arcomora.

Relay modules

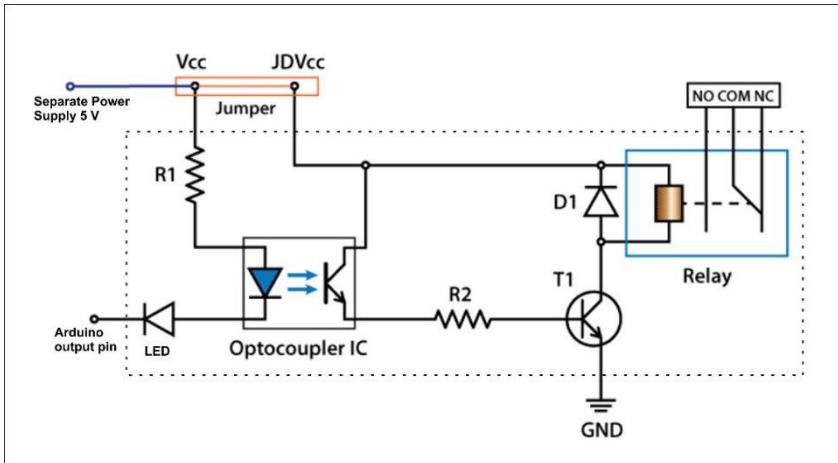
If you use the relay modules with optocoupler as mentioned on <http://www.arcomora.com>, it may be important to know that these are active LOW.

This means that the relay is energized when the voltage on the Arduino port is 0 volts.

There is a standard jumper over the ports Vcc and JDVcc.

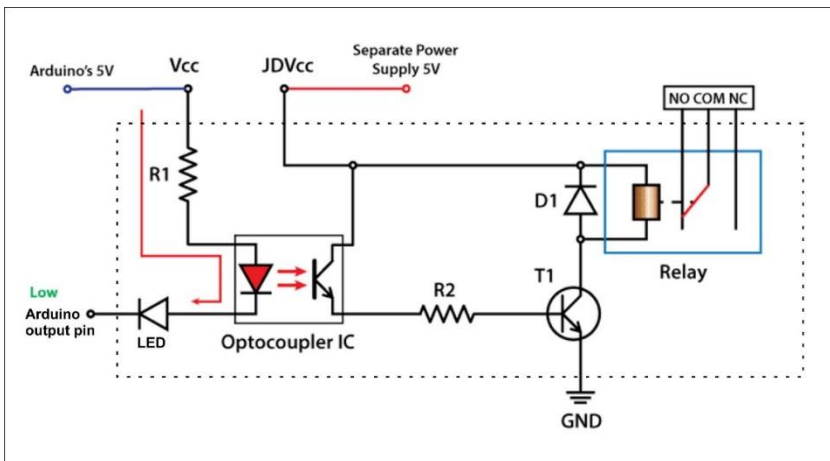
You now have two options:

1) Leave the jumper and connect Vcc / JDVcc to external power supply



2) Remove the jumper and:

- Connect Vcc to an Arduino 5V output
- Connect JDVcc to external power supply



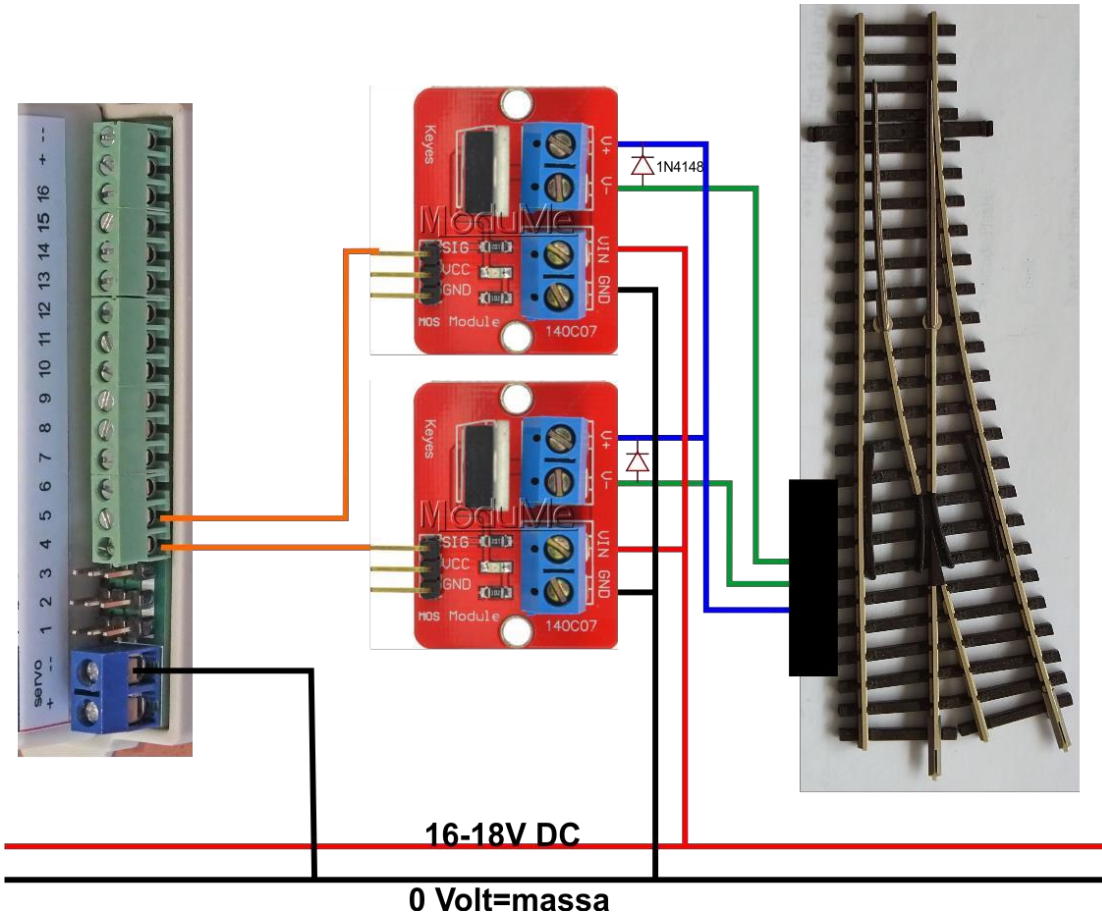
In this picture you also see that the relay is energized when the Arduino port is LOW.

In any case, you must NOT leave the jumper and connect Vcc / JDVcc to the Arduino.

In that case you also pull the coil current from the Arduino and that is not wise.

Sample configurations

Connecting a turnout with coils and two MOSFETS amplifiers



...

```
Enter command (P/A/D/E/I/R/?): P
Select port number.
Enter value from 1 to 16: 4
Port is undefined. First enter DCC address

Set DCC address for port 4
Enter value from 0 to 2000: 4
DCC Address set to 4

Select port type: Accessory(A), Servo(S) or Input(I): A
Port 4 set as Accessory port.

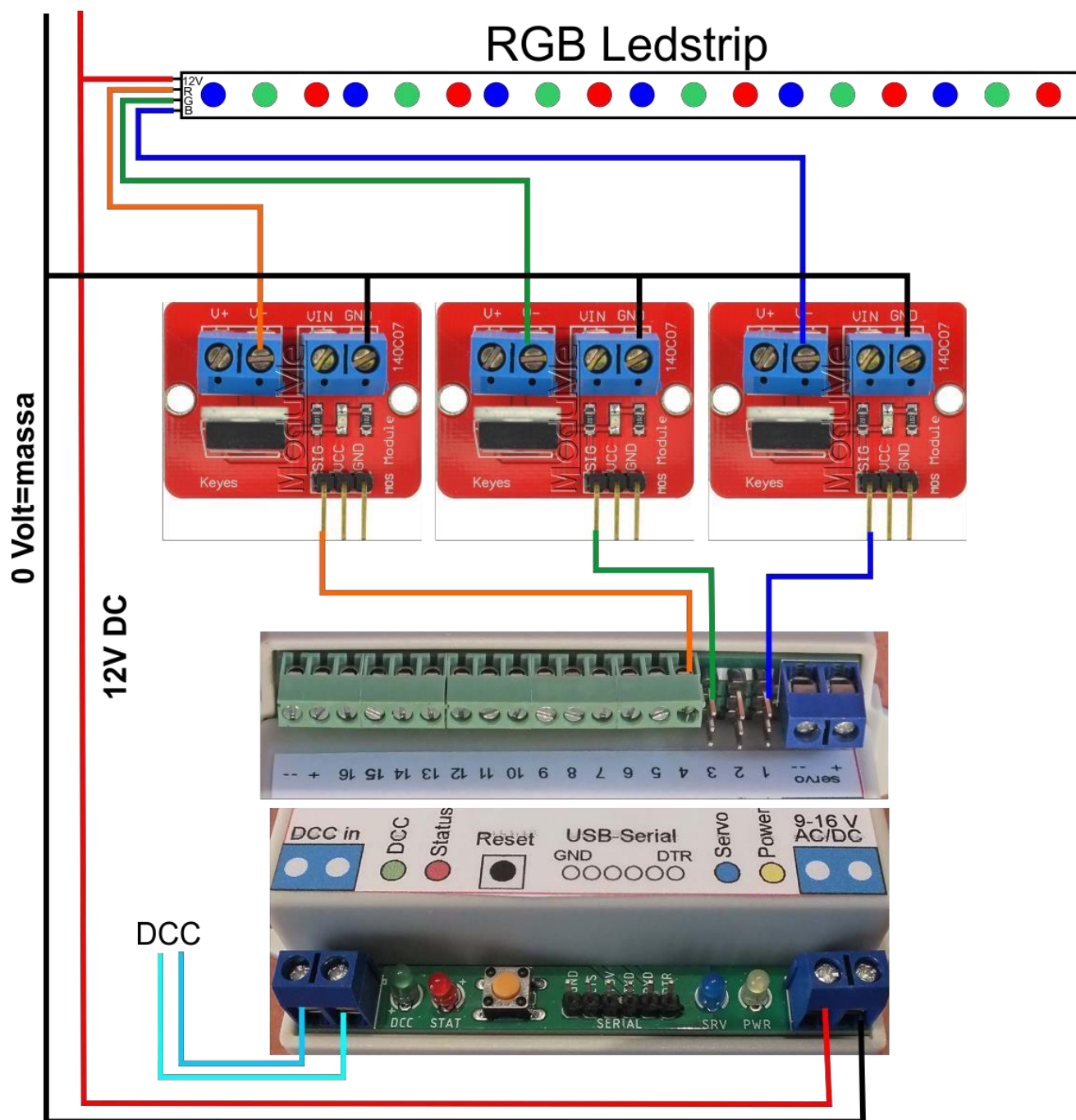
Select mode for this accessory
0=Help. Enter value from 0 to 9: 6
Mode set to Double One shot

Port 5 is assigned as buddy port to accessory on A4/P4
Enter 'On' time in milliseconds
Enter value from 5 to 30000: 250
'On' time set to 250 msec.

Enter command for Accessory on A4/P4
L/H/R/F/I/M/T/A/D/N/enter/?:

Port settings are saved!
```

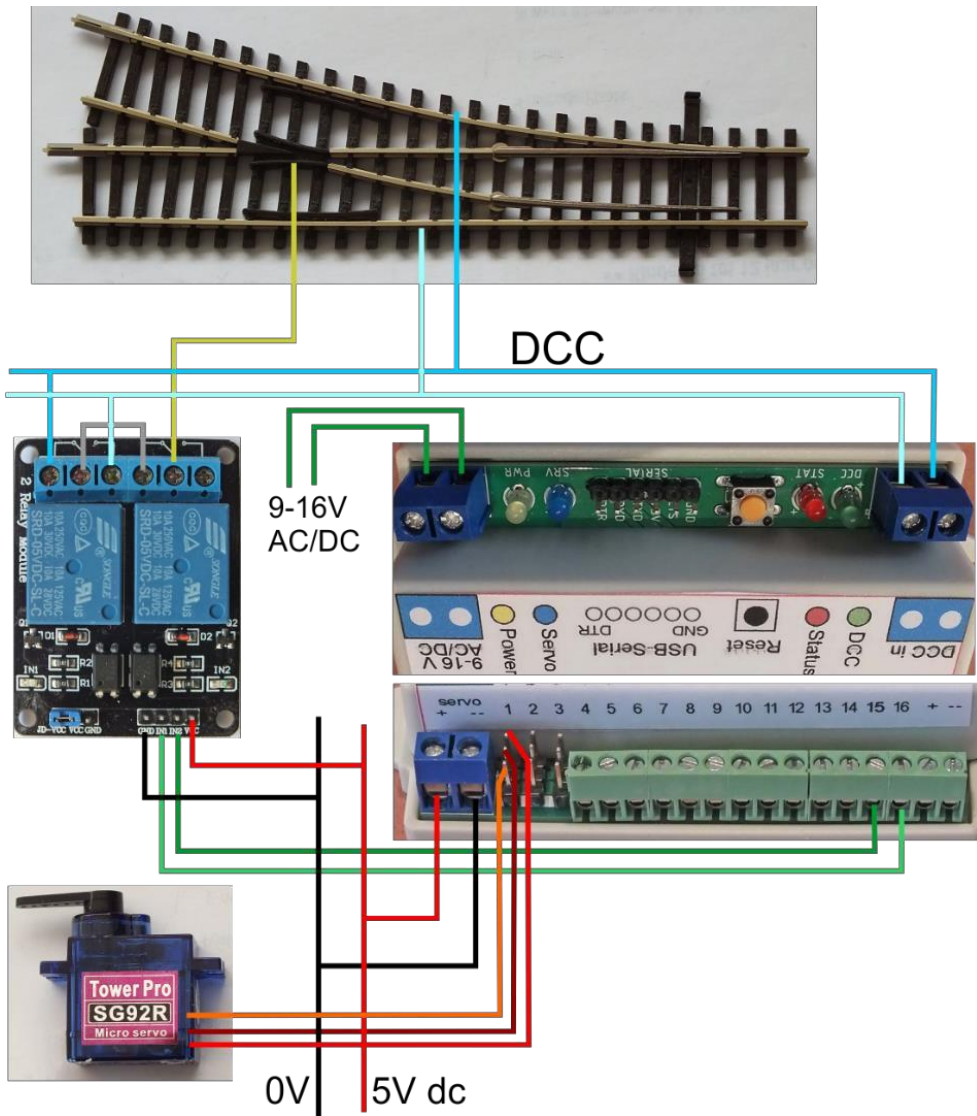
Connecting an RGB ledstrip with MOSFETS amplifiers



```

Port 1: DCC 12, Acc.type 7 (PWM), , Not Inv., PWM 5/150, Time R/F(sec) 800/800
Port 2: not used
Port 3: DCC 12, Acc.type 7 (PWM), , Not Inv., PWM 5/150, Time R/F(sec) 800/800
Port 4: DCC 12, Acc.type 7 (PWM), , Not Inv., PWM 5/150, Time R/F(sec) 800/800
Port 5: not used
    
```

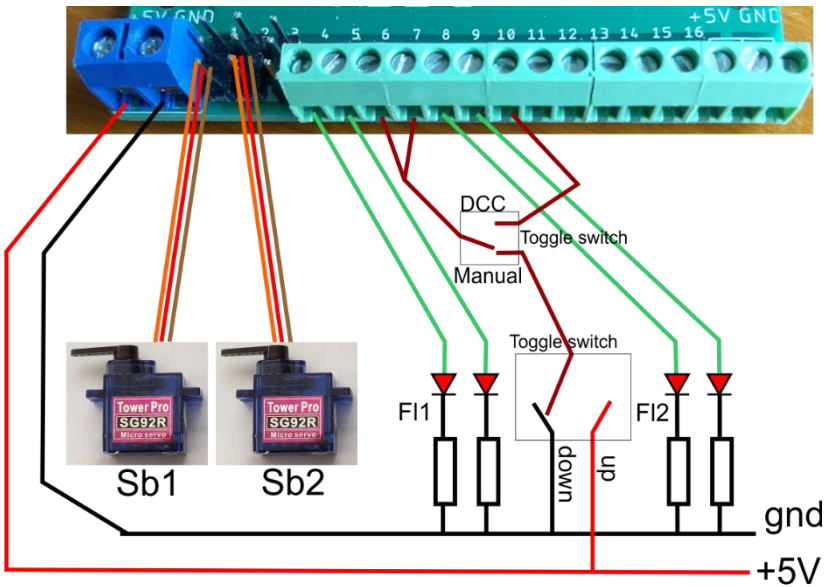

Connecting a servo and a double relay to a turnout



```

Enter command (P/A/D/E/I/R/?): p
Select port number.
Enter value from 1 to 16: 1
DCC 1, Servo Sb1, Angles 55/127, Not Inv., Speed 75, Frog port no, Bounce (L/H): 31/42
Angle set to HIGH: 127

Enter command for Servo on A1/P1
+/-/9/C/F/I/B/S/T/R/A/D/N/enter/?: F
How many relays? Enter value from 0 to 2: 2
Set Frog port. Enter value from 1 to 16: 15
Port 15 set as first frog port
Set second Frog port. Enter value from 1 to 16: 16
Port 16 set as second frog port
    
```

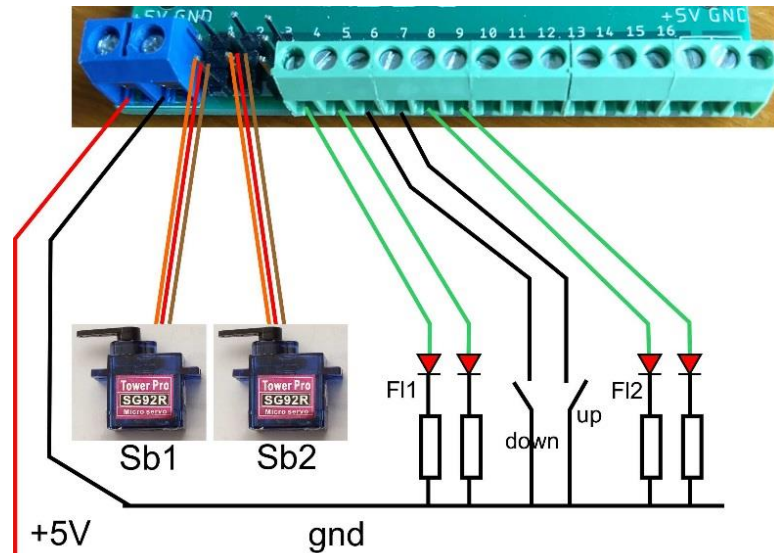


Railway crossing

Railway crossing controlled by DCC (single steady on port 10) or manual with two toggle switches.

```

Port 1: DCC 1, Servo Sb1, Angles 55/127, Not Inv., Speed 75, Frog port no, Bounce (L/H): 31/42
Port 2: DCC 1, Servo Sb2, Angles 53/131, Not Inv., Speed 75, Frog port no, Bounce (L/H): 42/31
Port 3: not configured
Port 4: DCC 4, Acc.type 4 (D. Flashing), F11, Not Inv., Buddy 5, Time(ms) 300/300
Port 5: Buddy for port 4
Port 6: DCC 4, Input down, Trigger: D, Second addr/delay: 1/3.0 sec, Third addr/delay: none
Port 7: DCC 1, Input up, Trigger: U, Second addr/delay: 4/7.0 sec, Third addr/delay: none
Port 8: DCC 4, Acc.type 4 (D. Flashing), F12, Not Inv., Buddy 9, Time(ms) 300/300
Port 9: Buddy for port 8
Port 10: DCC 2, Acc.type 1 (S. Steady), , Not Inv.
Port 11: not configured
Port 12: not configured
  
```



Railway crossing controlled manual with 2 switches. No DCC required

```

Port 1: DCC 1, Servo Sb1, Angles 55/127, Not Inv., Speed 75, Frog port no, Bounce (L/H): 31/42
Port 2: DCC 1, Servo Sb2, Angles 53/131, Not Inv., Speed 75, Frog port no, Bounce (L/H): 42/31
Port 3: not configured
Port 4: DCC 4, Acc.type 4 (D. Flashing), F11, Not Inv., Buddy 5, Time(ms) 300/300
Port 5: Buddy for port 4
Port 6: DCC 4, Input down, Trigger: D, Second addr/delay: 1/3.0 sec, Third addr/delay: none
Port 7: DCC 1, Input up, Trigger: D, Second addr/delay: 4/7.0 sec, Third addr/delay: none
Port 8: DCC 4, Acc.type 4 (D. Flashing), F12, Not Inv., Buddy 9, Time(ms) 300/300
Port 9: Buddy for port 8
Port 10: not configured
Port 11: not configured
Port 12: not configured
  
```


Release Notes version 6

NEW:

- The 'bouncing' of servos at the end of the rotation. The servo can thunder up to 4 times with angles of 1 to 9 degrees. This can be different for both angles.
- The test option for accessories has been changed. Instead of activating the accessory several times, the T-button can be used to turn the accessory on. This simulates the DCC signal.
- Buddy- and frogpoint ports can be chosen by yourself.
- The menu structure for changing accessories is simplified. Added the E-command; this is a combination of <enter> (=save) and the General E-command (=start of operating mode).
- The Single one shot can also be activated by a Down and/or an Up pulse.
- The two random functions are merged into one random function with on/off times from 20ms. to 600 sec.
It is also possible to set a fixed on or off time.
- An input now also has an inverted option.
- An individual port can now also be reset with the R command.
It is therefore no longer necessary to set the address to 0.
- Back to configuration state can now also be done with the P-command.
- The Quick-Config mode. This can be used to quickly make a change for a port.
With 'P' the configuration is started and the port number is requested immediately.
After the change you can immediately save the changes with 'E' and return to operating mode.

FIX: You can also use servo's on ports 12 – 16

NOTE: because the storage of the configuration in EEPROM is changed Mardec must be reconfigured after updating from 4.0 to 6.0