tcp184

User Manual Version 4

Table of Contents

www.bennypass.com

tcp184

Documentation history

v4.01   Major update to the underlying dScript, now with functions and local variables. Added "From Address" for easymail emails.

v4.02   Bug fixes for the underlying dScript.

v4.03   Editor update, added support for multiple editor tabs.
Compiler update, added support for multiple files in project.
Compiler update, added support for #ifdef, #else, #endif

v4.04   TCP server socket now transmits keepalives when connections become dormant for 10 seconds. 6 unacknowledged packets at 10 second intervals will result in port closure.

V4.05   Added phone app support (IO Network 2 on the App stores). Fixed bug that caused intermittent module not found error on firmware update.

V4.06   Added system reboot button to config status screen.
dScript updated to support dS2832. See dScript-v4.xx manual.

V4.07   Added new features:
Ping   Used to ping other machines to check if responding.
Sequencer   Added a 120 step, 12 output sequencer

v4.08   Added display of MAC address to network configuration page.
Bugfix – NTP (Real time clock) startup failed under some circumstances.
Renamed command "counters" to "setcounters".

V4.09   Added 3 new commands to Binary and AES Binary command set, to control all relays in one command. Set all, set selected and clear selected.

V4.10   Webpage security settings now on Network page for compatibility with other modules. The new dSx support for dS modules which have powered RS485 ports is not applicable to tcp184.

v4.11   Added Event Notifications, sends message to remote computer.

v4.12   Bugfix – Adding steps to the sequencer could crash the board.

V4.13   Bugfix – Notifications sometimes incorrectly reported Relay 1 changed.

V4.14   Bugfix – Flex input (used on dS378, dS2824, dS2832 & TCP184) notifications gave continuous outputs.

A quick look

Ethernet connected module, 10/100Mb auto negotiated.

Relays – 4 x 16Amp 250Vac C/O. with snubbers on N/O contacts.

Inputs – 8 optically isolated inputs, selectable as digital, 12-bit analogue or counter.

Power – AC mains universal input: 90 – 264Vac, 47-63Hz.

Connections – Screw Terminals for N/O N/C and Common contacts

Module size – 170mm x 104mm, DIN rail mounted

Controlled graphically by secure webpage or optionally one of ASCII, Binary or Modbus command sets over TCP/IP.

Email notifications – 8 selectable notifications, up to 100 emails/hour max.

Peer to Peer – Control relays on other modules.

Notifications – Sends a message when selected I/O change state.

Schedules – Able to schedule events based on time of day/week

Counter/Timers – Count or time events. Up to 2KHz counting on all inputs.

MODBUS gateway to downstream MODBUS modules such as the MBH88.

Introduction

The tcp184 is an Ethernet connected relay module featuring 4 channels of 16Amp 250Vac relays. Each relay has both normally open (NO) and normally closed (NC) as well as the common available on three terminals. It has snubbers fitted between the Common and N/O contacts.

In addition to the relays, the tcp184 has 8 input channels which may be individually configured to be:
1. Active low input with pull-up (allows direct connection of volt free contacts).
2. Active low input without pull ups (easy to connect logic level inputs).
3. Analogue inputs. 12-bit with selectable 0v-4.096v or 0v-5v input range.
4. High speed (2KHz) counters with or without pull-up resistors.

The tcp184 has four built in control methods.
Primary control method is graphically by using its built in secure website.
Secondary control may be one of:
ASCII - type in commands via a program such as PuTTY.
Binary - Command set using binary codes with optional AES encryption.
Modbus – Functions 1, 4, 5 and 15 supported.

And if you should wish to modify the supplied firmware – you can.
Behind the scenes there is dScript, a powerful multi-threaded operating system and programming language. The supplied firmware is written in dScript as well as HTML, CSS and Javascript on the webpages and the full source is in the separately down-loadable dScript support package.
You do not need to use dScript at all, but its great to know its there - just in case you do.

tcp184

Getting started

Start by wiring the mains input and plugging in the Ethernet cable to connect the module to your network. Switch on and the first thing you will note is that the blue LED will flash 3 times. This indicates that the control firmware is loaded on the module. (*If the blue led does not flash you will need to load in the control firmware. Don't worry, this is very easy to do. Just go to the chapter on installing the firmware and follow the instructions there*).

If you are using a Win10 PC, open your browser and into the address bar (not the search bar) type:
http://tcp184/index.htm

You should now see the application webpage and you can control relays and view inputs.

Relay 1	Relay 2	Relay 3	Relay 4
Relay 5	Relay 6	Relay 7	Relay 8
Relay 9	Relay 10	Relay 11	Relay 12
Relay 13	Relay 14	Relay 15	Relay 16
Relay 17	Relay 18	Relay 19	Relay 20
Relay 21	Relay 22	Relay 23	Relay 24
Relay 25	Relay 26	Relay 27	Relay 28
Relay 29	Relay 30	Relay 31	Relay 32

IO1 ⚫ IO2 ⚫ IO3 ⚫ IO4 ⚫
IO5 ⚫ IO6 🟢 IO7 ⚫ IO8 ⚫

| example.com | 0 | Ctr1 | 0 | | 0 |
192.168.1.2	0	Ctr2	0		0
	0	Ctr3	0		0
	0	Ctr4	0		0

tcp184

Locating the IP Address

If you are not using a windows PC then you will need to find the IP address of the module. The simplest method is connect the module (Ethernet, USB then lastly, Power) and load up the dScript editor.
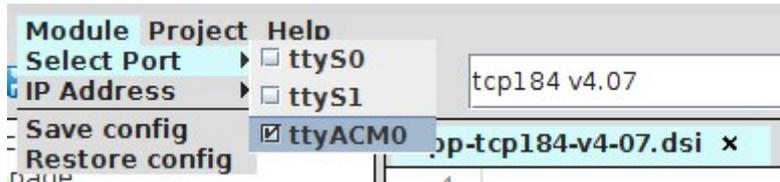
Go to Module→Select Port and make sure the correct serial port is selected as illustrated above.
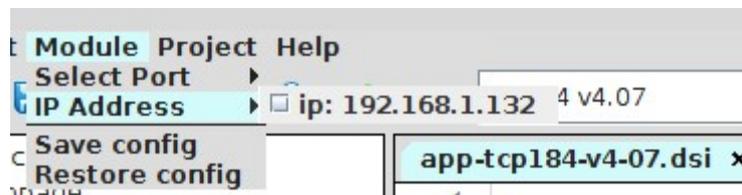
Now go to Module→ IP Address and the current IP address is displayed.

Make sure the Ethernet is connected before you apply power. If you plug the Ethernet cable in after the power, the module will already have booted with a default IP address.

Alternatively, you can find the IP address of the module by checking your DHCP server. If you have a DHCP server on your network (your router is normally the DHCP server) then the tcp184 will get its IP address from that. Log on to your router and navigate to the LAN client list.

Above is the entry from our ASUS RT-N66U router. So now you type:
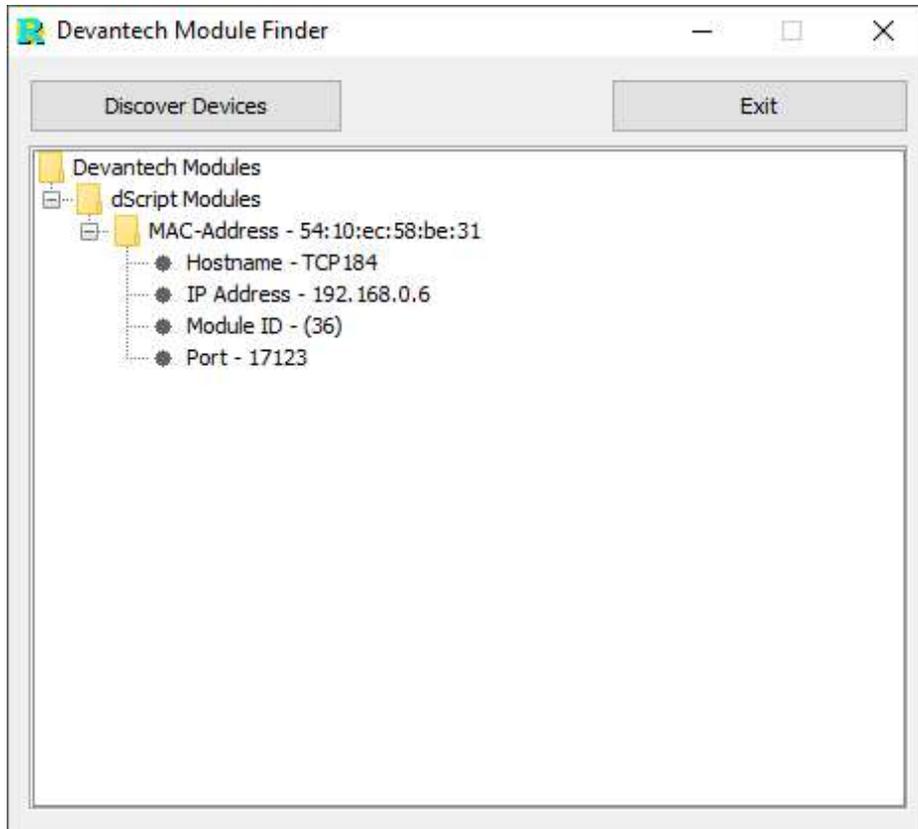192.168.0.6/index.htm
into your browser address bar and you should see the application webpage shown on the previous page.

tcp184

If you prefer, you can download a java program that will run on Windows, MAC or Linux, and will list all of our modules that are connected to your LAN. DevantechModuleFinder.jar

If you do not have a DHCP server the tcp184 will use a default IP address of 192.168.0.123 so make sure your PC is on the same subnet of 255.255.255.0 and its IP address is 192.168.0.xxx

www.bennypass.com

Configuring the tcp184

There are a set of configuration pages to get the tcp184 operating as you want it. These pages are all _configx.htm, (that's a leading underscore character).
ie.
_config.htm
_config2.htm

CFG/RUN link

Anything that starts with _config is considered a special name for configuration pages and can only be seen if you have the 3-pin CFG/RUN link in the CFG position. When the link is in the RUN position the _config pages are not accessible. This is a security measure. After you have configured and deployed the module, you really don't want these configuration pages available for others to change. So with the link in the RUN position the _config pages are not available. If you try to access them you just get served a "not authorised" page instead.

So for now, you do want to look over the config pages. Place the link in the CFG position.

With the link correctly positioned, browse to:
192.168.0.6/_config.htm
(substituting your IP address)

Status page

You should now see the following page:

This status page shows you the system and application firmware revisions.

If you hover your mouse cursor over the menu buttons on the left, the help panel will give you an overview of each one.

The system reboot button should be used with caution. If the module is at a remote site and you change the IP address, you may not be able to reach it again.

Network page

Notice that everything below the Host Name is greyed out and can't be changed. This is because the "Enable DHCP" box is checked and all the greyed out fields are supplied by the DHCP server. Although a quick way to get you connected, we really do not recommend this as the DHCP server can assign a different IP each time you power up. If you want to control this module from the internet while you are away from the premises then you will be setting up port forwarding on your router which requires a fixed IP address.

So lets do that first. Uncheck the DHCP box and you can then set all the other fields. Notice that the Red "Update Pending" light comes on. It indicates there are changes which have not yet been written to the flash memory. It will go off again 5 seconds after you stop changing anything.

Choose an IP address for the module, something outside of the DHCP settings on your router so it will not assign anything to that address. The subnet mask, Gateway and DNS can all be left as the defaults. Network changes only take effect after the next re-boot, so wait until the "Update Pending" light goes out and give the reset button on the module a quick press. The Green Led will light and the Blue led will flash 3 times. You will now find the module at your new IP address. If the Red led comes on after you press the reset button, its because you pressed it for too long (and entered bootloader mode). Just have another go with the reset button for a bit less time.

Your browser won't know you have changed the IP address so it will still be showing the old, now dead page. Make sure you change to your new IP address and load the page again.

---

www.bennypass.com

Webpage Security

Leaving the Security Password blank will disable it and allow everyone to access the application page to control the module.

To enable password protection enter a password into the Password box. You can use any characters from the ASCII character set from 0x20 to 0x7E except " (0x22). It may be up to 200 characters long and you don't have to memorise it, so make it a long one with plenty of uppercase, lowercase, numbers and symbols.

Accessing a special webpage, _pw.htm, will install the password on your browser. To do this make sure the Enable _pw.htm box is checked.

When the "Update Pending" light goes out, re-boot the module and go to yourIP/_pw.htm

You will see something like this:
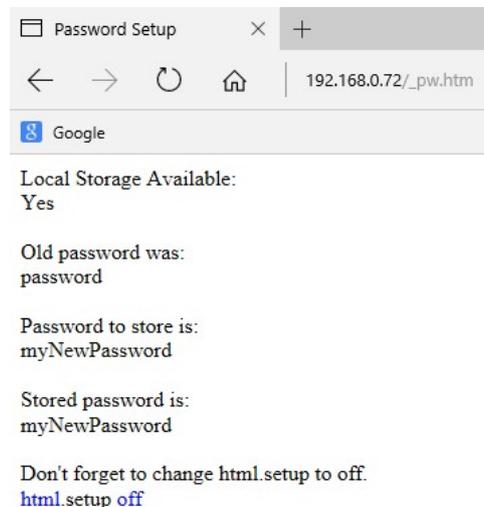
The password is now loaded on your browser. Do the same for any further browsers you want to enable. When you have done uncheck the "Enable _pw.htm" box to prevent anyone else from loading the password. When the "Update Pending" light goes out re-boot the module again.

tcp184

The default port used by html webpages is 80. You can change this if required. If you do so then you will need to include the port number in the address.
If you change the port to 2345 then the webpage will be at:
YourIP:2345/index.htm
For example:
192.168.0.123:2345/index.htm

www.bennypass.com

TCP/IP page

The TCP/IP tab allows you to select one of three command sets to control the module. These are independent of, and separate to the HTML webpage control.

Clicking on one of the four check boxes will select that command set. Only one command set may be selected. You can disable all TCP/IP command sets by clicking on an already selected box.

The ASCII command set is text based. You can use any program that can send text over TCP/IP. We use PuTTY in raw mode. The Binary command set operates in a similar way to our ETHxxx range of boards (but uses different commands), by sending binary commands to the module. AES Binary adds encryption. Note the AES key MUST be 32 bytes long. We have a test applications written in C# and Java to demonstrate AES encryption.

Phone App configures the module for control with our "IO network 2" app, available on iTunes/Play store. Password is an 8 character password used only with the App. Change this to a password of your choice and use the same password on your phone.

When selecting the Modbus commands, an additional set of configuration boxes are available.

These allow you to select the UID, normally you will leave this at the default of 1. The tcp184 will respond to commands on this UID. If you send any other UID it will be treated as the address of a Modbus module connected to the RS485 port. Commands to other UID's will be converted to Modbus RTU format and sent out on the RS485 port. Any responses will be converted back to Modbus/TCP format and returned to you.

The second box selects the baud rate of the Modbus RTU line and the third box selects even, odd or no parity.

Modbus command set implements functions 1, 4, 5 and 15.

When selecting the Modbus command set, make sure you select port 502 which is the official port number for Modbus over TCP/IP.

A full description of the commands is in the "Command Sets" section later in this manual.
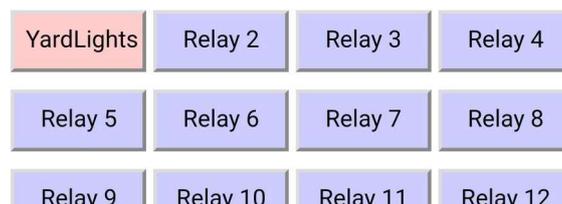
Naming relays

The next tab allows you to set the names of the relays that will be displayed on the application page.

Names may be up to 20 characters long, but keep them shorter if you want to use a phone as the controller. Here we have renamed relay 1 as "YardLights" and those 10 characters are about the limit on a vertical phone screen.

Relay automation

There are a set of control boxes that provide for autonomous relay operation. If you just want to control the relays from the webpage or using one of the TCP/IP modes, then leave these boxes blank.

Relay No. box
This selects the relay you wish to configure. There are 32 relays available. The first 4 relays are the actual relays on the module. Relays 5-32 are virtual relays. They have no physical presence but otherwise behave identically to the real ones. They can provide additional processing and a larger control surface to your application.

Pulse/Follow box
If you place a number in this box then that is the pulse time for the relay in mS (1000mS = 1 second). The minimum value is 100mS. The maximum is 2147483647 (more than 24 days). When the relay is toggled on, timing begins and at the end of the time period the relay goes off again. If the button is pressed again during timing then the timing is restarted.

The above example will turn off the yard lights after 30 minutes. 1800000 mS is 1800 seconds. 1800 seconds is 30 minutes.

It also possible to insert a boolean equation into the box. Boolean equations are just simple equations that evaluate to a 0 or 1.
For example Relay 1 is referred to a R1. Input1 is referred to as D1 (Digital 1). Analogue input 1 is referred to as A1.

tcp184

Here's a very simple example:

Enter R2 into the Relay 1 Pulse/Follow box. This will make relay 1 copy whatever you do to Relay 2. Try it!

Now change it to !R1.

The exclamation mark is read as "Not R1". Now relay 1 will always be the opposite of relay 2. Try it.

Boolean equations may be used for controlling relays and other objects such as email.
See the "Boolean equation" section later in this manual for full documentation.

Set Reset & Toggle boxes
When used, these three controls contain boolean equations. The "Set Relay" box will set the relay when the boolean equation becomes true. The other two boxes reset and toggle the relay when the boolean equation become true.

tcp184

This is important! The Set, Reset and Toggle controls are transitory (edge triggered) controls.

The relays are only affected at the moment the boolean equation becomes true. This differs from the Follow box where the relay continuously follows the output state of the equation.

In the previous example where we placed R2 in relay 1's Follow control, relay one will always be the same as relay 2. If relay 2 is off and we click relay 1 (YardLights) nothing will happen.

However if we want relay1 to be controlled by relay 2 as above, but also want to be able to toggle it on/off from the webpage, we can put R2 in the "Set Relay" box and !R2 in the "Reset Relay" box.

Relay No.	1 ▾
Relay Name	YardLights
Pulse/Follow	
Set Relay	R2
Reset Relay	!R2
Toggle Relay	
Power-up Restore	☐

Try it.

Power-up restore
When checked, the relay is restored to the previous state it had when power was lost. If the relay was on when power was lost or the module was turned off, then the relay will be turned on again when the module is powered up again.

Naming Input's

The I/O Names tab is used to assign meaningful names to the Input terminals.

As with relay names these may be up to 20 characters long, but do check it looks ok on a mobile device or whatever you are using to control the module.

The I/O types box allows you to select the type of I/O you want.

The two digital modes are the same except for "Digital No-Pullup" the passive pull-up resistor is turned off and for "Digital With Pullup" it is turned on. The no-pullup option is useful for connecting to digital circuits (3.3v or 5v) that supply the input voltage. The with pull-up option turns on a weak pull up to 5v. This is useful for directly connecting volt free contacts such as relay contacts or switches. They are connected between the pin (P) and ground (0v).

The analogue mode converts the input voltage to a digital number in the range 0-4095 (12-bit conversion). The input voltage range is selectable between 0-4.096v (Ref4) or 0-5v (Ref5).

Counter modes select the high speed counting feature. Each input can count at up to 2KHz and has selectable pull-up resistors.

When you have changed I/O types wait for the "Update Pending" light to go out and press the reset button, as new I/O modes are set on power-up.

Sequencer

The sequencer runs a continuous loop of up to 120 steps. At each step you can specify a time delay and the outputs. The K1 – K12 outputs are updated and the start of the delay time. The example above shows the simplest sequence possible. At line 01 it delays 1 second and turns on the K1 output. Line 02 delays 1 second and turns off the K1 output. Line 03 jumps back to line 1 and the cycle repeats forever.

The K1 – K12 outputs may be used in boolean equations to control relays, send emails, etc.

Placing K1 in a relays Pulse/Follow box will make the relay follow the K1 output, in this case pulsing the relay on and off every two seconds.

www.bennypass.com

Sequencer commands

There are a set of 12 commands to control the sequencer.

The simplest of these is just a number which is the time delay in seconds.

1234	Numbers may be up 4093 seconds.
m500	This is the time delay in mS. There is a limit of Max. 4093ms.
M10	The time delay in minutes. There is a limit of Max. 4093 minutes.
H2	The delay in hours. There is a limit of Max. 596 hours.
U90	Unspecified delay in seconds. Its a random number up to the limit specified.
	The Max. delay that may be specified is 4093 seconds.
T16:45	Waits until the specified time before continuing. Make sure you have an internet connection and the clock is correctly set up on the Scheduler page.

At the start of all the delay commands above, the K outputs will be updated with the specified settings, then after the delay time, the sequencer moves on to the next line.

None of the following commands will update the K outputs. If you specify anything in these check boxes they will be ignored. The instructions are executed immediately with no delays.

J1	Jumps to the specified line.
C9	Calls the specified line. The sequencer has a 4 deep call/return stack. The line following this one is pushed onto the stack and control transferred to the specified line.
X	eXit. Returns from a previous call. The line that was pushed onto the stack from the last call is popped off and control transferred to that line. There are no parameters to this command.

The following conditional jump commands take two parameters, the object number and the line to jump to. They will jump to the specified line if the object is active (on), otherwise continues with the next line. They do not stop and wait.

R1, 12	Tests R1 (Relay 1) and jumps to line 12 if active (on).
D1, 20	Tests D1 (Digital Input 1) and jumps to line 20 if active.
S1, 30	Tests S1 (Schedule 1) and jumps to line 30 if active.

K Outputs

The sequencer outputs are a set of 12 flags, K1 to K12. These may be used anywhere a boolean is used, to control relays etc.

For simple sequences the K output can be used in the Relays pulse/Follow box.

More complex sequences using call/return instructions should use the K flags in the Set, Reset and Toggle boxes. Then use one K flag to set the relay and another to reset it. The K flags are now only be needed at the point you need to switch the relay. In this way a common called sequence will only affect the specified relays, not everything.

Sequencer Examples

Scanning inputs, waiting for something to do.

Supposing you need to monitor two inputs and perform one of two control sequences depending on the input. We will need a loop to check the inputs and transfer to the appropriate sequence when active.

If D1 becomes active control transfers to line 10.

If D2 becomes active control transfers to line 20.

Otherwise they arrive at line 3 and jump back to line 1.

The sequence will loop around these three lines until one of the digital inputs becomes active.

At the end of the routines at lines 10 and 20 you should finish with jumps back to line 1.

Traffic Lights

Here in the UK the traffic light sequence is Red, Red + Amber, Green, Amber, Red. You would need more than two relays for this, but we can watch the sequence on the virtual relays on the application page. Here is a sequence for a 2-way traffic control of the type you normally see at road works.

Here we hold both lights at Red for 30 seconds to give traffic time to traverse the controlled area. Then Red + Amber for two seconds followed by Green for 60 seconds. We then stop the traffic with Amber for 3 seconds followed by Red. Again we hold both lights at Red for 30 seconds before letting the other queue through at line 6.

To watch it happen, set relays 21, 25 & 29 to K1, K2 & K3 and set relays 24, 28 & 32 to K4, k5 & K6.

Then look at the application page.

tcp184

Adding a pedestrian crossing to the sequence

In many sequences there are common runs of instructions. Our sequencer allows you to separate out these common sequences and call them from elsewhere in the sequence. Having only a single copy of a sequence aids in maintaining the program. We will expand our traffic light example to demonstrate the call and return instructions, by adding a pedestrian crossing.

We use a digital input for the request to cross button. Before changing our sequence code, we need to provide a latching request signal because pedestrians expect to give the request button a brief push, not hold it down. Lets use relay 22 for the latched request, D1 will set the relay and K7 will clear it.

Relay Name	X Request
Pulse/Follow	0
Set Relay	D1
Reset Relay	K7
Toggle Relay	
Power-up Restore	☐

Enter D1 in relay 22 Set Relay box and, as the relay will be reset by the sequencer, enter K7 in the Reset Relay box. Also enter K8 and K9 into relay 23 and relay 27 pulse/follow boxes. While we are in the relay section, lets rename the relays to better reflect their functionality.

(21) North RED	(22) X Request	(23) X RED	(24) South RED
(25) North AMBER	.	(27) X GREEN	(28) South AMBER
(29) North GREEN	.	.	(32) South GREEN

Now we can modify the sequencer code. Change the 30 second delays on lines 1 and 5 and replace them with C13. This is a call to line 13, which is on page 2. This will keep the pedestrian routine on one page for easier reading. Also remove all ticks from K1 – K12 on these lines. The Call instruction will ignore them anyway, but it's good to keep things clear.

Enter the following on page 2.

Line 13 is the 30 second delay where we hold all lights at Red. These are the two delays from

page 1 lines 1 and 5, where we replaced them with the C13 command. We still need this delay.

Line 14 tests the pedestrian request latch, relay 22, and jumps to line 16 if active, otherwise it continues with line 15 which just return to the caller on page 1.

Line 16 turns the pedestrian signal to Green for 15 seconds and also clears the request signal.

Line 17 returns the pedestrian signal to Red and waits a further 5 seconds for pedestrians to complete the crossing.

Line 18 returns to the main sequence on page 1.

Give it a try!

tcp184

Schedules

The scheduler can schedule regular events. These can be once or twice daily with the two start and stop times and can happen on any selected weekdays.

The Schedule No. is one of eight schedules that can be set up.

The target relay is the local relay that you want this schedule to control.

W/Days selects the days of the week you want this schedule to control the relay. When high-lighted in red, the weekday is active.

There are two sets of Start and Stop times for each of the eight schedules. The relay is set when the start time transitions from inactive to active, ie. at the start time, and reset when the stop time is reached. It is not held on or off. It can be changed by other events including the webpage buttons.

The current time originates from an internet time server. It is read only and cannot be changed. Therefore you must have an internet connection for the scheduler to operate correctly.

The timezone allows you to set the time for your location.

For example;
GMT leave this at 0.
CET set this to 1.
PST set this to -8
IST set this to 5:30

Daylight saving time may be checked if required. It advances the time by 1 hour between the last Sunday in March and the last Sunday in October.

As well as controlling a relay directly, schedules may also be used in the boolean equation fields of other controls. A relay could be controlled by placing S1 (for schedule 1) in its automation field.

tcp184

Counter/Timers

Count input pulses or time events.

There are a total of eight counter/timers available, selected with Counter No. box. Each counter can count digital inputs at a maximum speed of 20Hz (20 counts per second) or high speed counter inputs at a maximum speed of 2KHz (2000 counts per second).

Counter Name
Each counter/timer may be named and this name appears on the application page and email notifications.

Count Input
Each counter/timer is capable of being a timer by entering T1 into its count input box. T1 is an internal 1Hz timer that will advance the count once per second. If and input is specified then any pulses on that input will be counted. D4 will count pulses on input 4 at up to 20Hz. HSC4 will count at up to 2KHz. Make sure the input is correctly configured in the Input/output tab. Note that "HSC4" directs the system to use the high speed counter. HSC4 cannot be part of a boolean equation.

Capture Input
This is the event that will cause the current counter value to be stored in the capture register. The capture register is displayed on the application page, and may also be read using the TCP/IP commands.

---

www.bennypass.com

31

Reset Input
This input will reset the counter value to zero. If the capture Input has been left blank then it will store the current counter value in the capture register before resetting it to zero. You may use an input such as D3 or you can use the counter value itself. Entering C1>9 for the counter1 Reset Input will reset the counter when it reaches 10. This will create a repeating 10 step timer (0-9).

Counters, Timers and Schedules can be combined. As an example lets assume we want to count the pulses on input 2. We need to know the number of pules per hour coming in. First we need a 1 hour time base, we'll use counter/timer 1 for this.

Counter No.	1 ▾
Counter Name	Ctr1
Count Input	T1
Capture Input	
Reset Input	C1>3599
Current Value	17
Captured Value	0

Using T1 for the count input will make the counter increment once per second. There are 3600 seconds in 1 hour so we reset the counter when its greater than 3599 with C1>3599. The counter will count from 0-3599.

Next we will use counter 2 to do the counting of input 2 pulses.

Counter No.	2 ▾
Counter Name	Ctr2
Count Input	D2
Capture Input	
Reset Input	C1<1
Current Value	0
Captured Value	0

The count input is set to D2 to count the pulses. The reset input is controlled be counter 1. When it gets reset to zero each hour, so does counter 2. As the capture input is blank the reset event also captures the count value. This may be read any time in the next hour.

The internal time base, T1 is derived from the crystal on the module. It's accurate but will drift over time so that the capture event may not happen "on the hour". Even if you started it on the hour is will drift out by a few seconds a day. So as a final refinement we will synchronize our time base with the real time derived from a schedule.

The target relay is set to zero as we don't need to use any relays for this. All week days are set so we perform the synchronization every day. Start/stop time 1 is used to activate the schedule at 1 second past midnight each day. The second start/stop time is disabled as both times are identical.

Now go back to counter1 and add "|S1" to the reset input. Now our time base timer will reset every hour or when schedule1 triggers it.

www.bennypass.com

Event Notifications

Event notifications will send a tcp/ip message to a remote computer when selected Inputs or Relays change state.

Event triggers are boolean expressions. They may be a single or multiple I/O.

R1 will send a notification when Relay 1 becomes active.

{R1} will send a notification when Relay 1 changes state (to active or inactive).

Enclosing the boolean expression in curly braces {} will return true when the I/O changes state. To use multiple I/O, separate them with an | (or).

{D1|R2} will trigger a notification when either D1 or R2 changes state.

Target IP is the IP address of the computer you wish to send the message to.

Target Port is the tcp/ip port the computer is listening on.

Timeout is the time the board waits for a return message. This return message is used only as an ACK to terminate the connection, and may be a single byte.

TimeStamp, when checked, will include the time and date in the message.

The messages are in human readable plain text and are also easily readable in code.

```
41 1 D1
13/12/22 11:45:38
10000000000001100000000000000000
1 639 0 0 0 0 0 0
```

The first line informs you of the event number that triggered the notification and the state of that trigger, followed by the event in a human readable form.

Note that the on-board digital inputs D1-D8 are actually events 41-48. The relays are 1-32.

In this case digital input D1 became active.

The next line, if the TimeStamp box is checked, is the date (day/month/year) and the time (hr:min:sec).

This is followed by the 32 relay states reading left to right, 1 to 32. In the above example relays 1, 14 and 15 are active.

Next are the 8 input states reading left to right, 1 to 8. In this case input 1 is active and is the trigger for this event. It is configured as a digital input which will display as 0 or 1. Input 2 is configured to analogue mode and is showing its analogue value.

Node-RED

A simple Node-RED flow can be used to receive the notification.
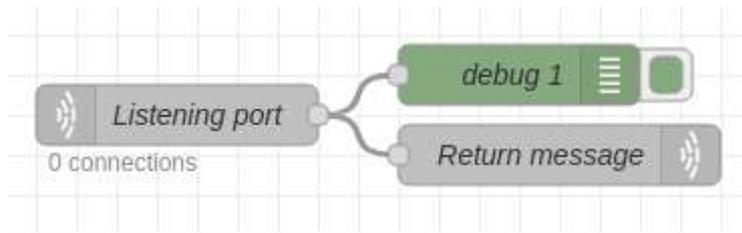
A tcp in node is connected to a tcp out node and a debug node. The tcp in node is set to listen on the selected port. The tcp out node is set to "Reply to TCP". The debug node allows you to see the message being sent. We used this setup to generate the message screenshot above.

See the "HomeAssistant-NodeRED" manual for more information and an example of extracting individual inputs from the message.

Peer to Peer

This tab allows you to configure events on this module to control relays on another.

Up to eight (8) Peer to Peer events may be set up, selected P2P No. box.

Control of the target relay is by using the binary command set only. This can optionally use AES encryption. The Target module must be set to either Binary or AES Binary mode on its TCP/IP tab, and this "Use AES" checkbox set to match.

The input box selects the event that will be used to control the target relay. This can be as simple as an input, or a complex Boolean equation. See the "Boolean equation" section later in this manual for full documentation.

Target IP, Port & Relay No should be set to locate the target relay.

Target action determines how the relay will be controlled.
0 – No action, this Peer to Peer event is disabled.
1 – Follow Me, the relay will follow the input, (or the Boolean result of any equation here).
2 – Set Relay, the relay will be set by this event. You will need another P2P event to reset it.
3 – Reset Relay, the relay will be reset by this event. You will need another P2P event to set it.
4 – Toggle Relay, this will toggle the relay each time the event is triggered.
    20+ - Pulse relay, Pulse times are in mS. (1000mS=1Second). 5-19 no action.

Email notifications

The Email tab is used for sending secure, AES encrypted email notifications from the module.

Up to eight (8) email notifications may be set up, selected by Email No. box.

Setting up emails is quick and easy. You just need the recipients email address, a notification message which will be the email subject line and the trigger event.

The trigger event uses the same boolean equation solver as the relay automation. So if you want to trigger an email when digital input 1 (I/O1) becomes active, just enter D1.

The email message is automatically filled in with useful information on the state of the relays, I/O's etc.

Boolean equations are used for controlling relays as well as triggering email notifications.

See the "Boolean equation" section later in this manual for full documentation.

Note - Email notifications are limited to 100/hour.

---

tcp184

Pinging remote machines

The ping module allows you to check other machines are still online and responding.

Ping threads are only started a boot time if the Repeat Time is greater than zero. Also DNS lookup is only performed once at boot time if the Repeat Time is greater than zero. So after making changes, wait for the Update Pending light to go out (5 seconds after last change) and reboot the module.

Repeat Time

This is how often the ping is repeated, in seconds. 10 sends a ping every 10 seconds.

Attempts

The number of failed pings (no response) before failure is reported by setting the ping time to zero. If set to 5 it will take 5 consecutive failed responses to trigger a failure. If a response is received, say on the 3rd try, the attempt counter is reset to 5.

Delay

This is the delay between detecting the failed responses and re-starting the pings. It is also the startup delay before beginning pings. Its purpose is to give the target machine time to boot up, or reboot.

Ping Time

The time between sending the ping message and receiving thee response. Timing is given in uS, but subject to a 1-2mS jitter.

Rebooting the target machine

A relay can be used to reboot the target machine. This may be by connecting to its reset input, or even removing and restoring power. The details of this are left to you.

In this example we are going to generate a reset pulse by pulsing Relay1 for 2 seconds when the ping time goes to zero. To do this enter 2000 in the pulse/follow box and P1<1 in the Set Relay box.

If you are going to reboot by removing power, you might want a longer pulse, say 30000mS or 30 seconds.

You might also send a notification email:

The application page

The last tab in the configuration pages takes you directly to the application page so you can quickly see the results of your configuration changes.

The I/O indicators show grey when the I/O is inactive. When an active input is applied they show green.

Application page security

The configuration pages (any page name starting with _config) are only served when the CFG/RUN link is in the CFG position If its in the RUN then no configuration pages are served. Instead, you will be served a page saying "You do not have permission to view this page."

The _pw.htm page (which contains the javascript that loads the password into your browser) will only be served when the enable _pw.htm checkbox is checked on the Webpage Security tab.

If the password field is left blank, the application page will always be served – to anyone! Entering a password means the application page will only be served to a browser that has the matching password set. Everyone else will just just be served a simple webpage saying "You do not have permission to view this page."

To summarize, set a password, load it onto your browser, disable _pw.htm and place the link in the RUN position. Now you, and only you, can access and control your module.

Accessing your webpage from the internet

Now you have your webpage up and running on your local network, for example 192.168.0.150, and you can access the webpage and control the module.
You just go to 192.168.0.150/index.htm, and the page is there.

However you can't get directly at that page from your phone when you are away from home. You can't access it on 192.168.0.150 because your network is not publicly accessible, its a private network address. You will have another IP address. This is the one your ISP gave you for your internet connection, and is the public IP address of your router on the internet. If you don't know what it is you can type "my ip" into Google's search bar and it will tell you. This is the IP address you will use to access the modules webpage.

Everything on the internet uses an IP address and a port number.
When you access a webpage in your browser all you enter is the IP address (or more likely a domain name, but its ultimately translated to an IP address). You don't normally have to enter a port number but its still required. Your browser simply uses the default port number, which is 80 for the web, unless otherwise specified our modules also use port 80 for the webpage. However its a good idea to use a different port number for our boards as this will avoid conflict with any web server you may have on your network.
Pick a number, I'll choose 19321 as our port number. Just make sure its different from any TCP/IP port number you are using. The HTML port is set on the Webpage Security tab.

After you have re-loaded the program you can access the webpage with:
192.168.0.150:19321/index.htm

Note that as we have changed the modules html port we need to tell the browser how to find the page with the new port number. Do that by inserting a ':' character and the port number between the IP address and the page name as shown above.

Assuming your routers internet IP address is 86.87.88.89 (I made that up – replace with your actual IP address) you will access the page from anywhere with address:
86.87.88.89:19321/index.htm

However first you have to set up your router to do that.
It's called "port forwarding" or "virtual server", but whatever your router calls it, you need to set it up so that all incoming connections on port 19321 are forwarded to port 19321 on local IP address 192.168.0.150.

Unfortunately there are so many routers out there we cannot give details on all of them. You should consult your routers manual or search Google for details on your specific router.

Boolean equations

Both relay automation and email triggering use the boolean equation solver.

The types that can be used in boolean equations are:
1. Relays, R1 – R4
2. Digital I/O's, D1 – D8
3. Analog inputs, A1 – A8
4. Pings P1 - P4
5. Sequencer, K1 - K12
6. Schedules, S1 – S8
7. Counters, C1 – C8
8. 1Hz Time base, T1

The simplest equation is R1. This is true when R1 is active and not true when R1 is in-active. If you enter R1 in the relay 2 automation box it will simply follow whatever R1 does.

The exclamation mark ! is used as a "not". So !R1 is true when relay 1 is in-active.
Enter !R1 in the relay 2 automation box it will follow the opposite of R1. Relay 2 will be active when relay 1 is inactive.

The same applies to the digital I/O's. Enter D2 in the relay 2 automation box and the relay will follow the input.

Analog inputs are compared with a value to obtain a true/false boolean result. In this example we have set I/O8 to be an analogue input with a 5v reference. Then we can enter A1<1000 in the relay2 automation box. This will turn on relay 2 when the input A1 falls below 1000. If A1 is connected to a temperature sensor and R2 controls a heater – well, you get the idea. Analog comparisons use the "less than" < and "greater than" > symbols only. There is no equal or not equal. Checking for equality on a potentially jittery analogue input is not really useful.

As well as "not" !, you can use "and" &, "or" |, "xor" ^ in your equations.
Enter D2&D3 and the result is true only when both D2 and D3 are active.
Enter D2|D3 and the result is true when either D2 or D3 is active.

Any inputs or relays enclosed in curly braces will detect a change of state, both to active and inactive. {D1|R1} will give a momentary output when either D1 or R1 changes state. This can be used to set or reset relays, or as a trigger for notifications.

What happens here:
D2|D3&D4
The answer is that boolean expressions are evaluated left to right. So D2 is ORed with D3 and the result ANDed with D4. You can change the order of precedence by using parenthesis ().
D2|(D3&D4)
will now AND D3 with D4 and the result is Ored with D2.

To demonstrate a real world example, take the analog example above where we compared A1 with 1000 to operate R2. Whilst this would work its a not a good solution as the relay would jitter badly when A1 was hovering between 999 and 1000. What we need is some hysteresis. To do that we will use R2 in its own equation.

(A1<1000&!R2)|(A1<1234&R2)

Assume R2 in inactive (off). The 2nd half of the equation (A1<1234&R2) will have no effect. So when A1 falls below 1000 the relay comes on. Now the 2nd half of the equation is true, and will stay true until A1 climbs above 1234.

So the relay becomes active when A1 is below 1000 and inactive above 1234.

We have hysteresis!

TCP/IP command sets

There are four TCP/IP command sets on four selectable check boxes, of which one or none may be selected on the TCP/IP config tab. These are ASCII, Modbus, Binary and Binary with AES256 encryption.

ASCII command set

The ASCII command set allows you to type commands using a TCP/IP terminal program such as PuTTY. Use PuTTy in raw mode.

Make sure you have you module set to ASCII mode and, after the "update Pending" indicator has gone out, re-booted the module so the changes take effect.

ASCII commands are all two character commands and are not case sensitive. Type ST followed by the return key. This is the STatus command. You will see:

Valid commands are:

ST   STatus request

SR [Relay number] [on/off] {Pulse time in mS}
SR 1 on   Set Relay 1 on
SR 1 off   Set Relay 1 off
SR 1 on 1000   Set Relay 1 on for 1 second

GR [Relay number]
GR 1   Get relay 1 status – responds with Active or Inactive

GI [Input number]
GI 1   Get Input 1 status – responds with Active or Inactive

GA [Analog Input number]
GA 1   Get Analog input 1 – responds with the 10-bit analog value

GC [Counter number]
GC 1   Get Counter1 – responds with count, capture values.

Note – on the tcp184, GI and GA are identical, getting the digital state or analogue value depending on the port configuration in the I/O Types tab.

Typical PuTTY session.

tcp184

Binary command set

Summary.
0x30   Get Status
0x31   Set relay
0x32   Unused (It's Set output on modules with bi-directional I/O line)
0x33   Get Relays
0x34   Get Inputs
0x35   Get Analogue
0x36   Get Counters
0x37   Update all Relays

0x30 (decimal 48) Get Status (1 byte command, returning 8 bytes)
This command returns 8 bytes of status data

Module ID	This will be 36 (0x24) for the tcp184
System Firmware Major	2 for example
System Firmware Minor	18 for example
Application Firmware Major	1 for example
Application Firmware Minor	2 for example
Volts	Power supply volts x 10. Example 125 is 12.5v
Internal Temperature (high byte)	x 10
Internal Temperature (low byte)	combined to 16 bits, 267 = 26.7 degrees C

In the above example the returned bytes would be:
0x24 0x02 0x12 0x01 0x02 0x7D 0x01 0x0B
The last two bytes combined are 0x10B which is 267 decimal.

0x31 0x02 0x01 0x00 0x00 0x00 0x00 Set Relay (7 byte command, returning 1 byte)
This command turns a relay on or off or pulses it for a time period and returns an ACK/NACK
byte. ACK=0, NACK=non-zero (actually the unknown relay number).
0x31   The actual command, the rest are parameters.
0x02   Relay number. Valid numbers are 1-8 (0x01-0x08)
0x01   Turn relay on (0x00 for off). This is ignored when following pulse time is >100.
0x00   } high byte   Pulse time
0x00   } mid high   These 4 bytes combined are a 32-bit pulse time for the relay
0x00   } mid low   when less than 100 (as it is here 0x00000000) its ignored
0x00   } low byte   When >100 this pulses relay on for that number of mS

To pulse relay 5 on for one second the command is:
0x31 0x05 0x00 0x00 0x00 0x03 0xE8

0x000003E8 (or just 0x3E8) is 1000 decimal, which is 1000mS or 1 second.
The relay will turn on and then go off 1 second later.
When sending a relay pulse time, the relay on/off byte is ignored. The relay is always on for
the duration of the pulse.

---

tcp184

0x33 0x01   **Get Relay (2 byte command – returning 5 bytes)**
This command is used to get the states of the relays. The second byte is the relay number, relay 1 in this case.
The first returned byte is the state of the requested relay, 0x00 (off) or 0x01 (on).
The next four bytes pack the states of all 32 relays (virtual and actual relays). Bit 7 of byte 2 is relay 32 through to bit 0 of byte 5 which is relay 1.

Byte 2								Byte 3								Byte 4								Byte 5							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

If the bit is set the relay is on, off otherwise.

0x34 0x01   **Get Input (2 byte command – returning 2 bytes)**
This command is used to get the states of the inputs. The second byte is the input number, input 1 in this case.
The first returned byte is the state of the requested input, 0x00 (inactive) or 0x01 (active)
the second byte packs the states of all 7 inputs. Bit 7 is unused. Bit 6 is input 7 through to bit 0 which is input 1. If the bit is high the input is active.

	Byte 2							
Bit No.	7	6	5	4	3	2	1	0
Input	x	7	6	5	4	3	2	1

0x35   **Get Analogue inputs (1 byte command returning 14 bytes)**
This returns all 7 possible analogue inputs. 14 bytes are returned, 2 for each analogue input.

Byte 1   byte 2   for example:
 0x02   0x3E   combined to 0x023E, or 574 decimal for input 1.

Bytes 3 – 12 follow in a similar way.

Byte 13 byte 14
 0x03   0x2C   combined to 0x032C, or 812 decimal for input 7.

If the input is configured as a digital port then the input will be 0 or 1.

0x36 0x01   **Get Counters (2 byte command – returning 8 bytes)**
This command is used to get the counters. The second byte is the counter number, counter 1 in this case.
The first 4 bytes returned is the current counter value. This 32-bit (4 bytes) value is returned high byte first. The second group of 4 bytes returned is the capture register for this counter, also a 32-bit (4 byte) value returned high byte first.

0x37 0xnn 0xnn 0xnn 0xnn   Update all relays (5 byte command returning 1 byte)
This command is used to update all relays with one command returning an ACK/NACK byte. ACK=0, NACK=non-zero. If a bit is high (1), the corresponding relay will be turned on. If a bit is low (0) the corresponding relay will be turned off. This command therefore affects all relays. The bit order is the same as the Get Relay command.

Byte 2									Byte 3									Byte 4									Byte 5							
7	6	5	4	3	2	1	0		7	6	5	4	3	2	1	0		7	6	5	4	3	2	1	0		7	6	5	4	3	2	1	0
32	31	30	29	28	27	26	25		24	23	22	21	20	19	18	17		16	15	14	13	12	11	10	9		8	7	6	5	4	3	2	1

0x38 0xnn 0xnn 0xnn 0xnn   Set selected relays (5 byte command returning 1 byte)
Set only those relays with a corresponding high in the bit pattern returning an ACK/NACK byte. ACK=0, NACK=non-zero. Same bit order as command 0x37 above.
Relays with a corresponding low (0) are NOT affected and remain on/off as previously set.

0x39 0xnn 0xnn 0xnn 0xnn Clear selected relays (5 byte command returning 1 byte)
Clears only those relays with a corresponding high in the bit pattern returning an ACK/NACK byte. ACK=0, NACK=non-zero. Same bit order as command 0x37 above.
Relays with a corresponding low (0) are NOT affected and remain on/off as previously set.

www.bennypass.com

AES binary command set

The AES Binary commands are the same as the Binary commands described above. The only difference is that they are AES encrypted and always 16 bytes in length. The first bytes are the same as described in the Binary command set. The last 4 bytes is the Nonce (a random number) and the bytes in the middle are undefined. The module will decrypt the command, generate the response and encrypt it before returning it to you.

Your program that controls the module will need to encrypt the commands and then decrypt the response.

We use AES256 CBC encryption, hence the requirement for a 256 bit (or 32 byte) key. To complete the security we use a random IV generated by a cryptoghraphically secure random number generator (ISAAC).

To control the module you will need to send the commands with AES encryption. To help you with this we have examples in C#, Java and Python. The C# and Java applications are complete and may be used or modified as you wish.

To prevent re-play (or Playback) attacks the command packet includes a Nonce. This takes the form of a 32-bit (4 byte) random number in positions 12, 13, 14 & 15 of the 16 byte data packet. For example when you send a Get Status command (0x30) you will get a 16 byte block returned. The first 8 bytes (0-7) will be as defined for the binary commands. Bytes 8-11 are unused. Bytes 12-15 contain the Nounce.

Commands which generate and send you a Nounce are:

0x30 – Get Status
0x31 – Set Relay
0x37 – Update all relays, On or Off
0x38 – Set only relays with corresponding bit set to On
0x39 – Clear only relays with corresponding bit set to Off

Commands which require a Nounce to be sent by you are:
0x31 – Set Relay
0x37 – Update all relays, On or Off
0x38 – Set only relays with corresponding bit set to On
0x39 – Clear only relays with corresponding bit set to Off

A Nounce is only ever used once, you must always use the most recently issued Nounce.

tcp184

The following example shows how the Nounce provided by the module is used in the next Set Relay command.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Get Status	0x30	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx
Response	0x23	0x02	0x12	0x01	0x02	0x7D	0x01	0x0B	xx	xx	xx	xx	0x89	0xAB	0xCD	0xEF
Set Relay	0x31	0x02	0x01	0x00	0x00	0x00	0x00	xx	xx	xx	xx	xx	0x89	0xAB	0xCD	0xEF
Response	0x00	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	0x1A	0x2B	0x3C	0x4D
Set Relay	0x31	0x03	0x01	0x00	0x00	0x00	0x00	xx	xx	xx	xx	xx	0x1A	0x2B	0x3C	0x4D
Response	0x00	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	xx	0xF1	0xE2	0xD3	0xC4

If the Nounce you send with the command does not match the last one sent to you, then the Relay (or Output) will not be changed.

No other commands either require or provide a Nounce.

Modbus commands

The modbus command set accepts a subset of the standard Modbus-TCP frames as defined in
[Modbus protocol Specification](#) and
[MODBUS Messaging on TCP/IP Implementation Guide V1.0b](#)

Functions 1, 4, 5 & 15 are supported along with error codes 1, 2 & 3 should they occur.

Function 01 (0x01) Read Coils

This function reads back the 32 relays as coils 1-32 and also the 8 inputs as coils 41-48.

If you do a block read from 1 to 48, the missing coils 33-40 will read as zero and not give an error.

Function 04 (0x04) Read Input Registers

This function reads back the 8 analogue inputs as registers 1-8

Registers 9-24 are 16 registers representing 8 counter values. As modbus registers are 16-bits wide and the counters are 32-bit, a counter is stored in 2 16-bit registers, high word first. In the same way, the 16 registers 25-40 hold the 8 capture values.

Register 9	Counter1 high word	Register 25	Capture1 high word
Register 10	Counter1 low word	Register 26	Capture1 low word
Register 11	Counter2 high word	Register 27	Capture2 high word
Register 12	Counter2 low word	Register 28	Capture2 low word
Register 13	Counter3 high word	Register 29	Capture3 high word
Register 14	Counter3 low word	Register 30	Capture3 low word
Register 15	Counter4 high word	Register 31	Capture4 high word
Register 16	Counter4 low word	Register 32	Capture4 low word
Register 17	Counter5 high word	Register 33	Capture5 high word
Register 18	Counter5 low word	Register 34	Capture5 low word
Register 19	Counter6 high word	Register 35	Capture6 high word
Register 20	Counter6 low word	Register 36	Capture6 low word
Register 21	Counter7 high word	Register 37	Capture7 high word
Register 22	Counter7 low word	Register 38	Capture7 low word
Register 23	Counter8 high word	Register 38	Capture8 high word
Register 24	Counter8 low word	Register 40	Capture8 low word

Function 05 (0x05) Write Single Coil

This function is used to write to a single coil (relay). Coils 1-32 are the relays 1-32.

Function 15 (0x0F) Write Multiple Coils

This function is used to write to multiple coils (relays). Coils 1-32 are the relays 1-32.

Error code 1

This error is returned if an unknown or unimplemented function is received. Only functions 1, 4 & 5 are implemented.

Error code 2

This error is returned if an illegal address is requested. Addresses greater than 15 for functions 1 & 5 or greater than 7 for function 4 will generate this error.

Error code 3

This error is returned if an illegal data value is received. Number of points = zero will generate this as will an illegal data value for function 5.

Modbus Gateway

If the Modbus/TCP/IP frame has a UID that does not match the one you set in the TCP/IP config screen, the module will assume this is for another Modbus module connected to the RS485 port. It will use the UID as the Module address, convert the request to RTU format and send it to the downstream module. This could be an MBH88 for example. The return frame is then converted back to Modbus/TCP/IP format and returned to you.

Make sure you set the Parity and Baud Rate in the TCP/IP config screen to match your Modbus RTU modules.

Loading the application firmware

If your module does not flash the blue LED three times on power-up, you will need to update the system firmware and load the application program. These instructions should also be followed if you want to update your firmware to the latest version or even revert to an earlier version.

You need to download the dScript programming environment from here:
http://www.robot-electronics.co.uk/dscript.html

PC requirements
Windows 7 or later, Linux or macOS.
USB port to program the module.

The dScript IDE is supplied as a zip file that can be download and unzipped into a temporary folder, inside the temporary folder will be four folders:

Installation
USBdriver
Examples
Documentation

Go to the Installation directory and click "setup" to install the dScript IDE, if you have already installed a previous version you will need to uninstall it before installing the new one.

The USBdriver folder contains the USB com port driver for the modules.

Copy the Examples directory to a convenient location on your computer, it contains both dScript source code examples and associated web pages, one of which is the application firmware you will need.

In this order:

1. Start from this position:
   a. dScript Editor closed down.
   b. tcp184 not connected or powered.

2. Power-up the tcp184.

3. Hold down the reset button for a couple of seconds until the red LED comes on. This indicates the module is in boot-loader mode.

4. Connect the USB lead to the PC. If windows wants a driver, point it to the USB driver folder and install the driver from there.

5. Run the dScript editor. Look in Help→About and check you have the latest version of dScript. In this case 4.12

6. If you have an earlier version you should uninstall it and install the new version from the installation folder, then start these instructions from the beginning. Older versions will not work with the new application.

7. Now look in the Module panel, you should see:
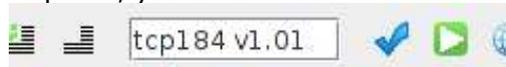
v1.01 indicates it's the boot-loader that is running.

8. Load the project: File→Open project
\dScriptPublish-4-12/Examples/app-tcp184-v4-12/app-tcp184-v4-12.dsj

9. Click the build button (white triangle on green button). This will update the system firmware and load the application.

10. When done, the new version will be displayed:

Erasing old configuration settings

Uploading the firmware, as described in the previous section, will not erase all the configuration values. If you need to clear these, do the following:

Load up the app-tcp184-v4-12.dsj application in the editor, but before you upload it to the tcp184 you need to make a small change. Locate the thread "main" (click the word in the right panel is quickest). Just below this is a commented out line "init()".

Uncomment this by removing the semicolon.

Now upload the application and it will reset the IP address along with all other variables. Confirm the board is operating, but don't change anything yet.
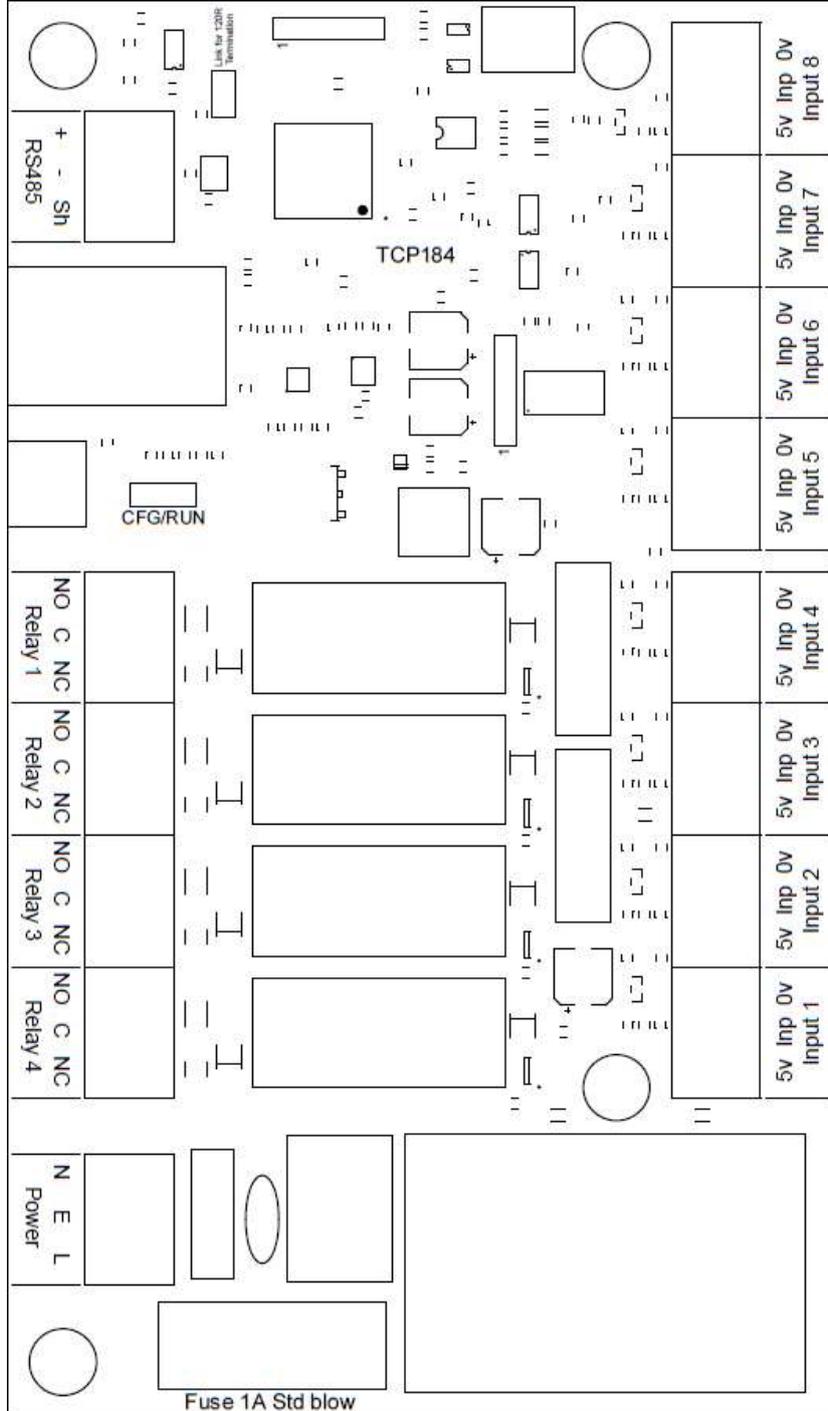
You need to replace the semicolon and upload the application again, otherwise it will continue to reset everything each time you power-up.

A faster method is to load the supplied Blank.dsf configuration file from the Examples folder.

tcp184 PCB

tcp184

The tcp184 provides four (4) volt free contact relay outputs with a current rating of up to 16Amp each, 8 flexible Inputs which can be analogue or digital or high speed counters and an RS485 serial port. The module has a universal AC mains input of 90 – 264Vac, 47-63Hz. The relays are SPCO (Single Pole Change Over) types. The normally open, normally closed and common pins are all available on the screw terminals. There is a snubber between the N/O and Common contacts.

LED indication

The tcp184 provides a row of three LEDs, Blue, Green and Red, are available for status indication. The Red LED lights when the module is in bootloader mode – this is when the IDE is uploading system firmware to the module. The Green LED lights the board begins running user programs. The Blue LED flashes 3 times on power up to indicate that the default app is running. All three LEDs are available and can be controlled as digitalports 33-35.

```
digitalport LedBlue      33
digitalport LedGreen     34
digitalport LedRed       35
```

The "FlashingLeds" example provides a colourful display showing how to use them.

Power supply

AC mains input of 90 – 264Vac, 47-63Hz.
Note. The Earth terminal is connected to the Input 0v terminals.

Operating temperature

-20C to +70C

---

www.bennypass.com
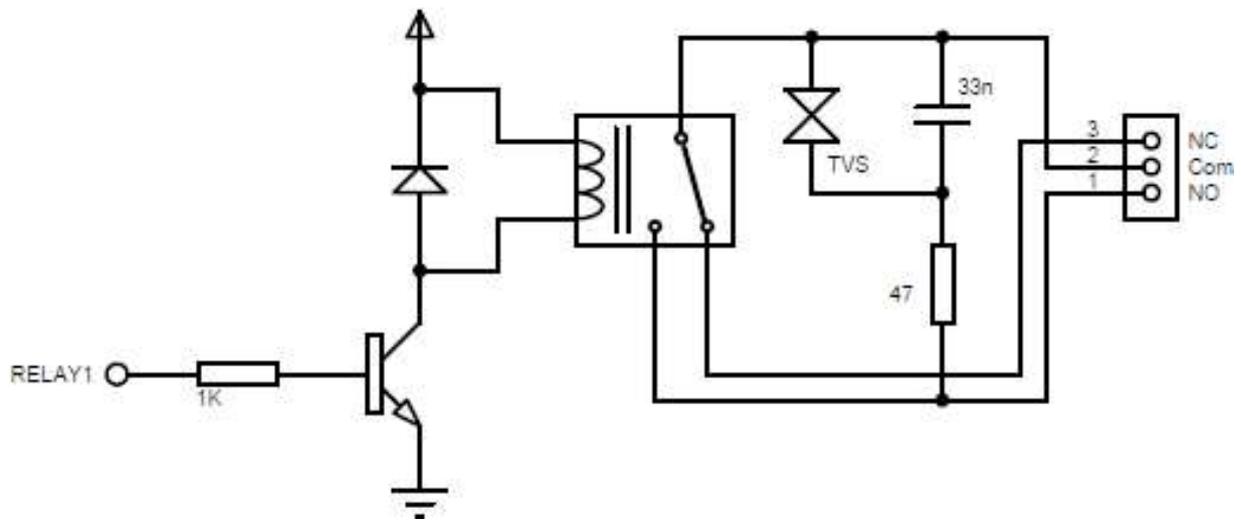
Power relays

Four16A volt free contact relays are provided for switching a common input between a normally closed output and a normally open output.

A full datasheet for the relays used on the tcp184 is here: HF115FD datasheet


The representative relay circuit used on the tcp184 showing the snubber components between the Normally Open (NO) and Common contacts.

Analogue/Digital flexible Inputs

The tcp184 has eight flexible Input ports, numbered 1-8, which can be your selection of:

1. Digital input with pull-up resistor (allows direct connection of Volt Free Contact, VFC).

2. Digital input without no pull-up resistor, suitable for driving with a 0-5v signal.

3. Analogue input using a 4.096v reference. 12-bit.

4. Analogue input using a 5v reference. 12-bit.

5. High speed counter, up to 2000 counts/second
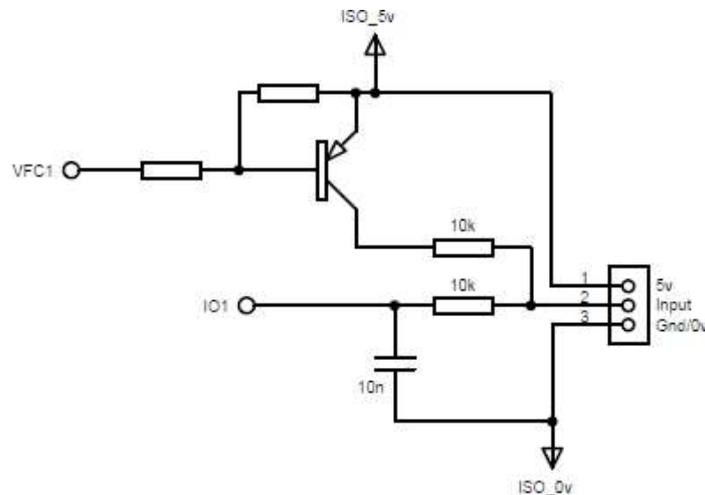
Representative input circuit.
The transistor is used to turn on/off the 10k pull-up resistor.
Another 10k resistor and 10n capacitor provides an input filter.

RS485 Serial Port Connection

The RS485 serial port is used by our default app to provide a Modbus Gateway.
When programming in dScript it is freely available for your own purpose.
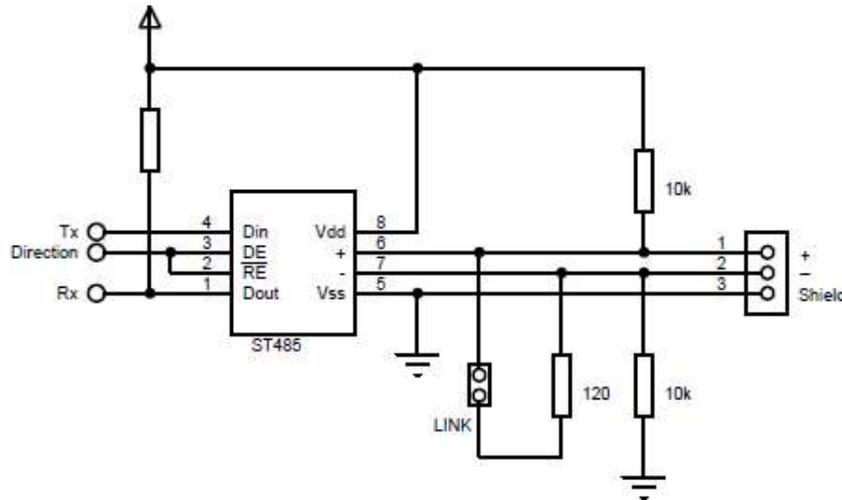
A two pin link near to the RS485 terminal block should be shorted to use the on-board 120 ohm terminating resistor.

Notes