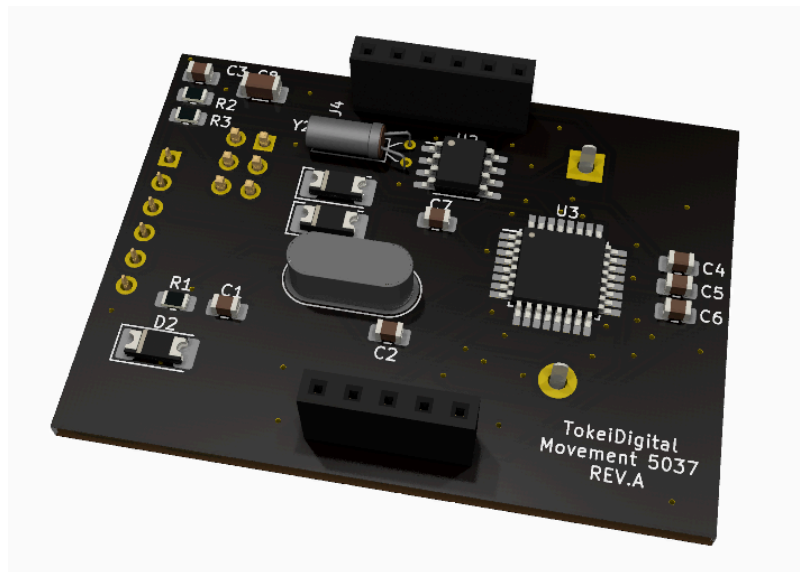


**TD-M5037A-FW1.0**



**User Manual**



<b>General</b>	<b>5</b>
Features	5
Connections	6
Setting Date and Time	9
Display Test Mode	9
<b>Assembling the Board</b>	<b>11</b>
Packing List	11
Top Assembly	12
Bottom Assembly	13
<b>Advanced</b>	<b>15</b>
UART	15
Physical Interface	15
Commands	16
Registers Map	20
Flashing the Arduino Bootloader	22
Flashing the Firmware	24
<b>Appendix A - HW Revisions</b>	<b>25</b>
<b>Appendix B - Schematic</b>	<b>27</b>



# General

## Features

The Tokei Digital TD-M5037 is a general purpose clock module that can be used as the base for designing clocks with different faces. It provides a **RTC**, with a backup battery, that can keep track of the current date and time even when the clock is not powered.

The module has a **UART** interface, allowing the user to set the time via a command interface and to tweak the configuration of both the module and the clock face.

Additionally the module provides several **GPIO** pins to both control a clock face and receive input from buttons placed on the clock face. These buttons allow the user to set the time in a more traditional way.

The module can also automatically switch to DST and back (EU rules only in FW1.0, for other territories DST can be set manually through the UART or simply ignored and time adjusted manually on the change date).

If you purchased the TD-M5037A as a kit to assemble see the "Assembling the Board" chapter for assembly instructions.

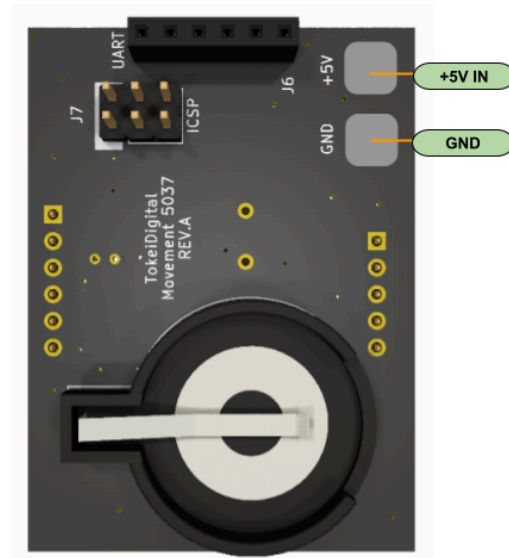
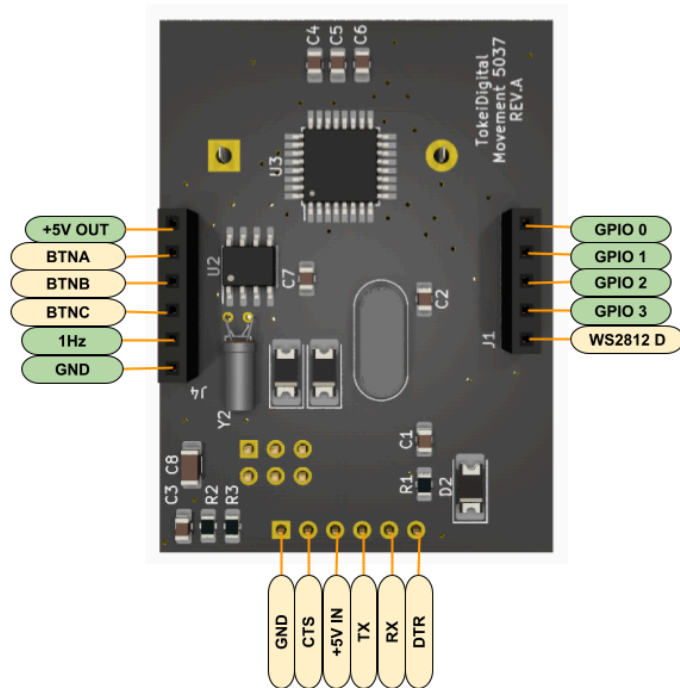
## Connections

Two female pin headers, protruding on the top side (opposite the battery) provide the mating interface to the clock face. The headers have 5 and 6 pins respectively to serve as a guide for the correct orientation of the module when plugging it on the face. These pins provide power to the clock face, digital data to control it, and bring into the module the lines from the three A,B,C buttons.

The choice to have the buttons on the clock face instead of the module was made to provide more flexibility in the design of the final clock, leaving freedom on the placement of the buttons on the face. In the same way, the power input to this module is made of two large pads where to solder the power supply. This allows for greater flexibility in the placement of eventual battery holders or external power connectors. An onboard USB connector would have instead limited the options and, depending on the clock face shape and orientation, could have resulted in the USB connector sitting in a non suitable location.

On the side opposite the headers is a holder for a CR2032 battery. If the battery is not installed time will be lost every time the power is lost. It's strongly recommended to use a backup battery, however normal operation is possible without.

On the same side is also an ICSP (In Circuit Serial Programmer) interface that allows to flash a new version of the firmware or to flash an Arduino bootloader. **If you received the module as part of a clock kit the firmware has already been flashed with the correct one for the face you selected and you will not need this interface.**



The table below details all the pins and their function.

+5V OUT	Power supply output for the clock face. Max available current for the face is 100mA.	CTS/TX/RX/DTR	UART interface. Allows to configure the clock through a serial interface. Also, if the module is flashed with the Arduino bootloader, sketches can be uploaded through this interface.
BTNA,B,C	Input pins for Buttons A,B,C. The module has internal pull-ups and expects the buttons to be pulling towards ground when pressed. These buttons allow you to do things like setting the date/time.	GPIO0,1,2,3	General input/output pins that can be used by the clock face.
1Hz	1Hz Square Wave output. This is an open collector output.	WS2812D	Dedicated output pin for clocks using WS2812 LEDs. If your clock face doesn't use such LEDs this pin can be used as general IO.
GND	Common ground connection for power to the clock face.	+5V IN	Power Supply pad. There is polarity protection on board but <b>no voltage protection</b> . Ensure your power supply is stable and <b>doesn't exceed 5V</b> . It needs to be able to supply enough current for the clock face (the movement consumption is few mA only).



## Setting Date and Time

**Time set mode is entered** with a **long press** on the **A** button. The first settable value (usually the hours) will start to flash on the clock face. Usually the clock will also indicate in some other way that it entered the time set mode, for instance flashing an LED or changing its color. For details see your specific clock face user manual.

**NOTE:** depending on how your clock displays time if the current value being set is zero you might not see anything flashing (e.g. in a binary clock). Press B or C once so the value is increased/decreased.

Once time set mode is entered use **B and C to increase/decrease** the value being set respectively. **To move to the next value** (e.g. minutes) just **press once A**. Depending on the abilities of your clock, you will cycle through all the values that can be set and return to the hours.

Once you are done setting the time, **long press** again **A** to **exit time set mode**.

**Hint:** to achieve a second-accurate setting, set the clock to the next minute. Then wait in time set mode until your reference clock shows :59 seconds, only at this point long press on A to exit time set mode. This works because, upon exciting time set mode, the clock always starts from second :00.

## Display Test Mode

It's possible to **enter test mode** for your clock by **long pressing** on the **C** button. In this mode your clock will light all LEDs regardless of current time. This can be useful to ensure all LEDs are functioning and to test your power supply and ensure it can provide enough current. **Long press** again **C** to **exit test mode**.



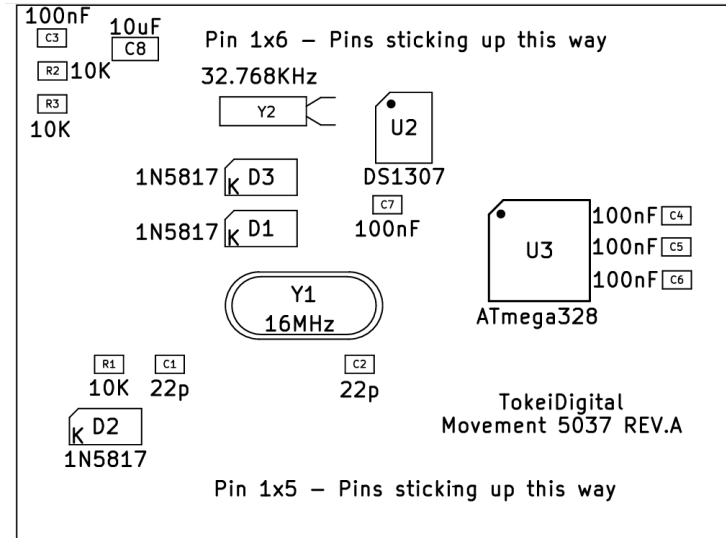
# Assembling the Board

## Packing List

Before proceeding with the assembly of the components on the board, double check the packing list to make sure you have received all components and that everything is in good order.

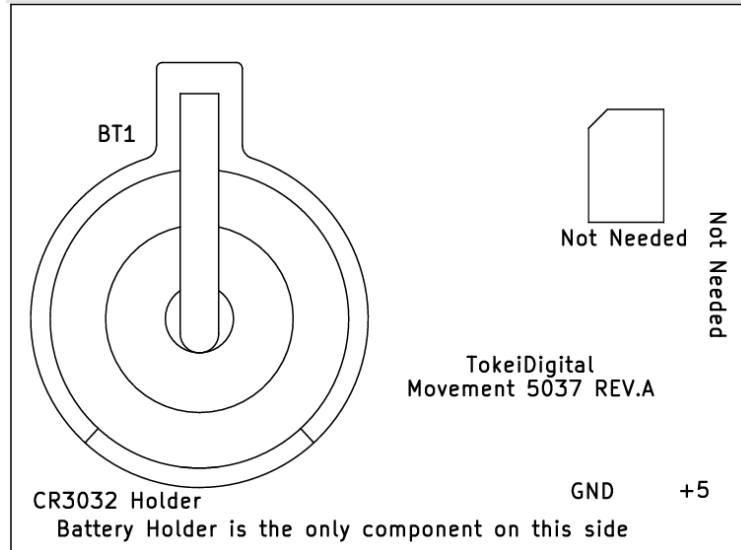
Bag	Content
A	2 x 22p (0805)
B	5 x 100nF (0805)
C	1 x 10uF (1206)
D	1 x 1N5817 (2010)
E	3 x 10K (0805)
F	1 x ATmega328 (QFP-32) 1 x DS1307 (SOIC-8)
G	1 x 16MHz Crystal 1 x 32.768KHz Crystal 1 x CR3032 Holder 1 x Pin 1x5 Vertical Female 1 x Pin 1x6 Vertical Female
	1 x PCB TD-M5037A REV.A

## Top Assembly



All components, with the exception of the battery holder, are assembled on the top part of the board. Depending on the tools you have available and your preference, you can use solder paste and position all the SMDs first to then reflow the board or just assemble the SMDs by hand. If you choose to assemble by hand make sure you have a very fine solder tip and very fine (0.2mm) tin. Pay attention to the orientation of diodes (the K marking on the image above corresponds to the grey band on the body of the component). Also the two chips need to be aligned with the dot on the body matching the image above. Assemble the crystals next and, finally, solder the pin headers on this same side of the board.

## Bottom Assembly



Flip the board and install the battery holder. For clock operation no other components are needed. If you are planning to use the UART you can install a 6 pin header on the pads labelled UART on the board. The choice of the type of pin header (male/female, 90 degrees etc.) depends on the type of USB-UART adapter board you have, whether you plan to encase the clock or not. The square pin next to the "UART" text is GND.

Also the ICSP connector (2x3 pins) is not needed unless you want to replace the firmware of the module, for instance to flash your own code for the clock face, an Arduino bootloader or an updated

version of the firmware. Again, the type of pins depends on your programmer cable, though usually these are male pins.

Finally a power source needs to be soldered to the GND and +5 pads. Once more, the type of cable, battery holder or perhaps USB connector adapter board depends on how you plan to display and eventually encase your clock. For this reason the board was not designed with a USB onboard connector but just with the pads.

# Advanced

## UART

The module exposes a UART interface that allows the user to configure the module, including setting time, DST and, depending on the clock face, color of the LEDs and other face features.

### Physical Interface

The only pins needed to control the UART are TX, RX, and GND. The other standard serial port lines are used only by the Arduino bootloader if you choose to flash it.

The UART operates at 9600 BAUD, 8 bits, 1 stop bit (no parity).

The terminal should be configured for CR+LF line ending.

## Commands

Press enter once you connect to get a list of all the available commands and a banner reporting the current hardware and firmware versions. If you don't receive any response, double check the connections and make sure the serial port parameters are set correctly.

TOKEI DIGITAL MOVEMENT 5037

VER: V1.0

FACE DRIVER: BUHR

H	HELP
N	PRINT NOW
R [START] [END]	DUMP REGISTERS
S [START]	SET REGISTERS
T H:M WD D-M-Y	SET DATE/TIME
X	EXIT
OK	

When you connect you might start to receive a burst of messages every second. These report the current date/time and other information. They can be set on/off by setting the relevant register (see next section). If the messages are coming they will stop once you press enter and will not resume until you exit interactive mode by giving the "X" command.



#### Print Current Time (N)

Send the "N" (now) command and press enter. The module will respond with the current date and time:

```
19:14 2 11-02-25
```

```
OK
```

The string is in the format:

```
HH:MM dayOfWeek day-month-year
```

Day of week is 1-7, with 1 indicating Monday.

#### Set Current Time (T)

Send the "T" (time) command followed by a space and the date and time you want to set. Use the same format returned by the N command.

```
T 19:14 2 11-02-25
```

```
OK
```

## Dump Registers (R)

Send the "R" (read registers) command and press enter. The module will respond with a dump of all registers. Optionally you can specify a start and end (hex values) to limit the output.

```
R 0B 14
0008      00.00.00.00.00
0010  00.00.00.00.00.
OK
```

```
R
0000  00.00.00.01.00.00.00.00
0008  00.00.00.00.00.00.00.00
0010  00.00.00.00.00.00.00.00
0018  00.00.00.00.00.00.00.00
0020  00.00.00.00.00.00.00.00
0028  00.00.00.00.00.00.00.00
0030  00.00.00.00.00.00.00.00
0038  00.00.00.00.00.00.00.00
0040  00.00.00.05.00.00.05.05
0048  00.04.02.00.00.00.05.05
0050  00.05.02.04.04.01.06.02
0058  01.05.00.02.02.02.30.30
0060  30.00.00.00.00.00.00.00
0068  00.00.00.00.00.00.00.00
0070  00.00.00.00.00.00.00.00
0078  00.00.00.00.00.00.00.00
OK
```

For the meaning of each register see the next section which details the movement registers. See also your clock guide for face specific registers.

## Set Registers (S)

Send the "S" (set registers) command followed by a space and the address (hexadecimal) of the first register you want to set. This command is interactive and will print the address and current value of the register.

You can now enter a new value, in hexadecimal and press enter. If you don't want to change that register value you can just press enter to move to the next. Once you are done editing press X and enter to exit the command.

```
S 7B
007B 00 .
007C 01 .01
007D 02 .02
007E 00 .X
OK
```

## Exit Interactive Mode (X)

Once you are done you can send the X command and press enter to exit the interactive mode. If the serial output is enabled (see register 0x03) the notifications will resume.

## Registers Map

0x00 - 0x3F Movement Registers	0x00	DST.	0=AUTO, 1=OFF, 2=ON
	0x01	Reserved for future use.	
	0x02	Reserved for future use.	
	0x03	Serial output protocol.	0=OFF, 1=V1
0x40 - 0x7F Face Registers	See face manual for usage.		

**0x00** If set to 0 DST AUTO is enabled and the clock will transition automatically in and out of summer time on the correct date. Currently the module supports the EU rules only. If you live in a different territory you can set manually DST on/off on the change date and the time will adjust. You can also leave DST OFF and change the time manually on the change day.

**0x03** Controls the serial output protocol. When enabled (set to 1 for Version 1) the module will send out messages at one second interval. The messages continue indefinitely until you press enter, at that point they stop to allow usage of the interactive interface. The messages will resume exiting interactive mode with the X command. Messages look as follows:

```
$V1!INFO0,V1.0,DCF77
$V1!DATE,11-02-2025
$V1!TIME,19:37:29
$V1!DOW,2
$V1!DST,OFF,AUTO
```

Each message starts with \$ followed by the protocol version. Following are comma separated tokens. The first indicating the message type and the others, in variable amount, are the parameters. Each message ends with a CR+LF.

**\$V1!INFO0** reports firmware version and face type.

**\$V1!DATE** reports the current date in dd-mm-yyyy format.

**\$V1!TIME** reports the current time in HH:MM:SS format.

**\$V1!DOW** reports the current day of week. The value ranges from 1 to 7. Monday being 1.

**\$V1!DST** reports the current DST status. The first parameter is either ON or OFF. The second parameter is present only if AUTO is enabled.

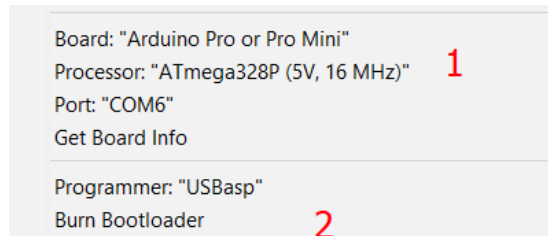
## Flashing the Arduino Bootloader

Whether you purchased the module on its own to design your own face, or you received it as part of a clock kit and you want to write your own firmware for it, you have the option to flash the Arduino bootloader on the board and use Arduino sketches.

**NOTE:** if you flash the Arduino bootloader it will override the clock firmware. For your clock to work you will need to write a sketch for it. Tokei Digital doesn't provide Arduino sketches for its clocks. However you can also flash back the correct firmware, see next section for details.

If you are sure you want to flash the Arduino bootloader follow these simple steps:

- Connect an ICSP programmer (such as USPasp) to the ICSP connector
- Open the Arduino IDE and configure the board as seen below (1). It's important you choose ATmega328P 5V, 16MHz or things won't work correctly.



- Choose "Burn Bootloader" (2)
- Once the process ends disconnect the programmer
- Connect now the board with a USB UART as any other Arduino board
- Select the right COM port that gets assigned to your board
- Upload your first sketch!

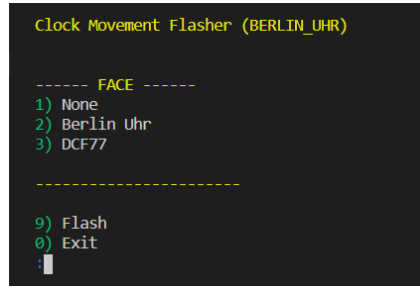
**Note:** REV.A of the board has a 100nF capacitor labelled C4 that interferes with the reset circuitry of the Arduino (which uses the serial interface DTR line to reset the processor). Remove this capacitor if you want to flash the Arduino bootloader. Future revisions of the board will omit this component as, in fact, it's not needed even in other configurations.

## Flashing the Firmware

If you purchased the module as part of a clock kit you do not need to flash the firmware as the ATmega was already flashed for you with the FW variant needed for your clock face. However, if you are planning to flash your own firmware, or restore the original one after flashing the Arduino bootloader follow these simple steps.

This assumes you have a USBasp ICSP programmer and a working installation of avr-gcc.

- Connect the board to the ICSP programmer
- Clone the <https://github.com/nicolacimmino/TokeiDigital> repository
- Open a unix like shell (e.g. GitBash) or your terminal if you are running linux
- Run the interactive flash tool you can find under src/movement with **sh flash.sh**
- You will be presented with a dialog like below:



```
Clock Movement Flasher (BERLIN_UHR)

----- FACE -----
1) None
2) Berlin Uhr
3) DCF77

-----

9) Flash
0) Exit
:|
```

- Select the correct face and then press 9 to flash

If you have a different programmer have a look into the Makefile in the src/movement folder of the repo and adjust for your programmer.



## Appendix A - HW Revisions

REV.A	First version. Only known issue at the time of writing is C4 interfering with Arduino reset. You can safely remove C4.
-------	--



