PIC-P14   development board

Users Manual

INTRODUCTION:

PIC-P14 board is development board which allow you to prototype and develop code for all Microchip's PIC microcontrollers with 14 pins.

What you can do with PIC-P14? A lot of stuff let's see what we have:

PIC-P14 have RS232 connector and driver, so you can interface your PIC microcontroller to other embedded or PC with RS232. The RS232 driver have 4 lines Tx, Rx, CTS and jumper selectable RTS or DTR. The pin labeled RX is output and should be connected to PIC UART Tx. The pin labeled TX is input and should be connected to PIC UART Rx. Note that MAX232 driver on the socket is 5V device, so if you decide to run your PIC on 3.3V you should replace it with MAX3232 which works on 3.3V

The on-board ICSP connector allow you to program the PIC on the board without pulling it of the socket, by ICSP programmer like PIC-MCP, PIC-MCP-USB, PIC-PG1, PIC-PG2, PIC-PG3, PIC-PG4 or to program and debug it with PIC-ICD2, PIC-ICD2-POCKET or PIC-ICD2-TINY. IMPORTANT: all programmers provide power supply through ICSP connector during the programming PIC-P18 should not be powered via the external power jack!

The power supply circuit have diode bridge after the power supply jack which allow you to power this board with both AC and DC power supply adapters. The input voltage could be anything within the range +7.5 – 15V DC or 6-12V AC.

Voltage regulator is made with LM317 adjustable voltage regulator and have jumper which selects what voltage you supply to the PIC microcontroller +3.3V or +5V, so you can work with PICs on both low and normal voltage.

The oscillator circuit is made with 20 Mhz crystal oscillator, so you can run your PIC at maximum performance.

The RESET circuit is made with simple RC circuit and RESET push button, the reason to not use RESET IC is that the board should support both 3.3V and 5V operation while RESET ICs are on fixed voltage.

PIC-P14 have user button for user input connected to PIC microcontroller's RC3 port. When RC3 port is initialized as INPUT you will read "0" when the button is pressed and "1" when it is depressed.

Status LED is connected via jumper to PIC microcontroller's RC2 port. When your RC2 port is initialized as OUTPUT and set to "0" LED will go ON, when RC2 port is set to "1" LED will go OFF.

PIC-P14 have handy GND pin for connection to oscilloscope.

FEATURES:
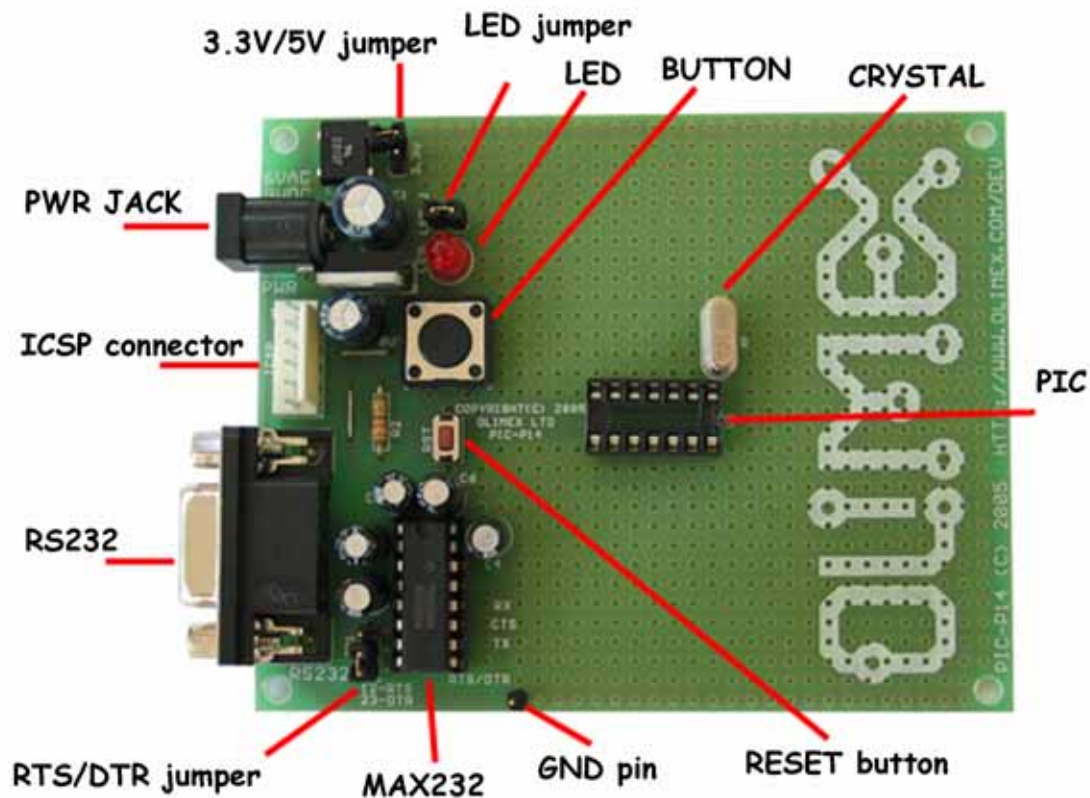
- ICSP/ICD connector for programming and debugging
- RS232 interface with MAX232 IC on socket
- DIL14 microcontroller socket
- Quartz crystal 20Mhz
- LED to RC2 through jumper
- user Button to RC3
- Reset button and circuit
- Power plug-in jack, accept AC and DC input
- selectable +3.3V / +5V power supply voltage regulator (3.3V when jumper is closed)
- Extension slot on every uC pin
- Gird 100 mils
- GND bus
- Vcc bus
- Four mounting holes 3,3 mm (0,13")
- FR-4, 1.5 mm (0,062"), green soldermask, white silkscreen component print
- Dimensions 100x80 mm (3,9x3,15")

 All boards produced by Olimex are ROHS compliant

HARDWARE:
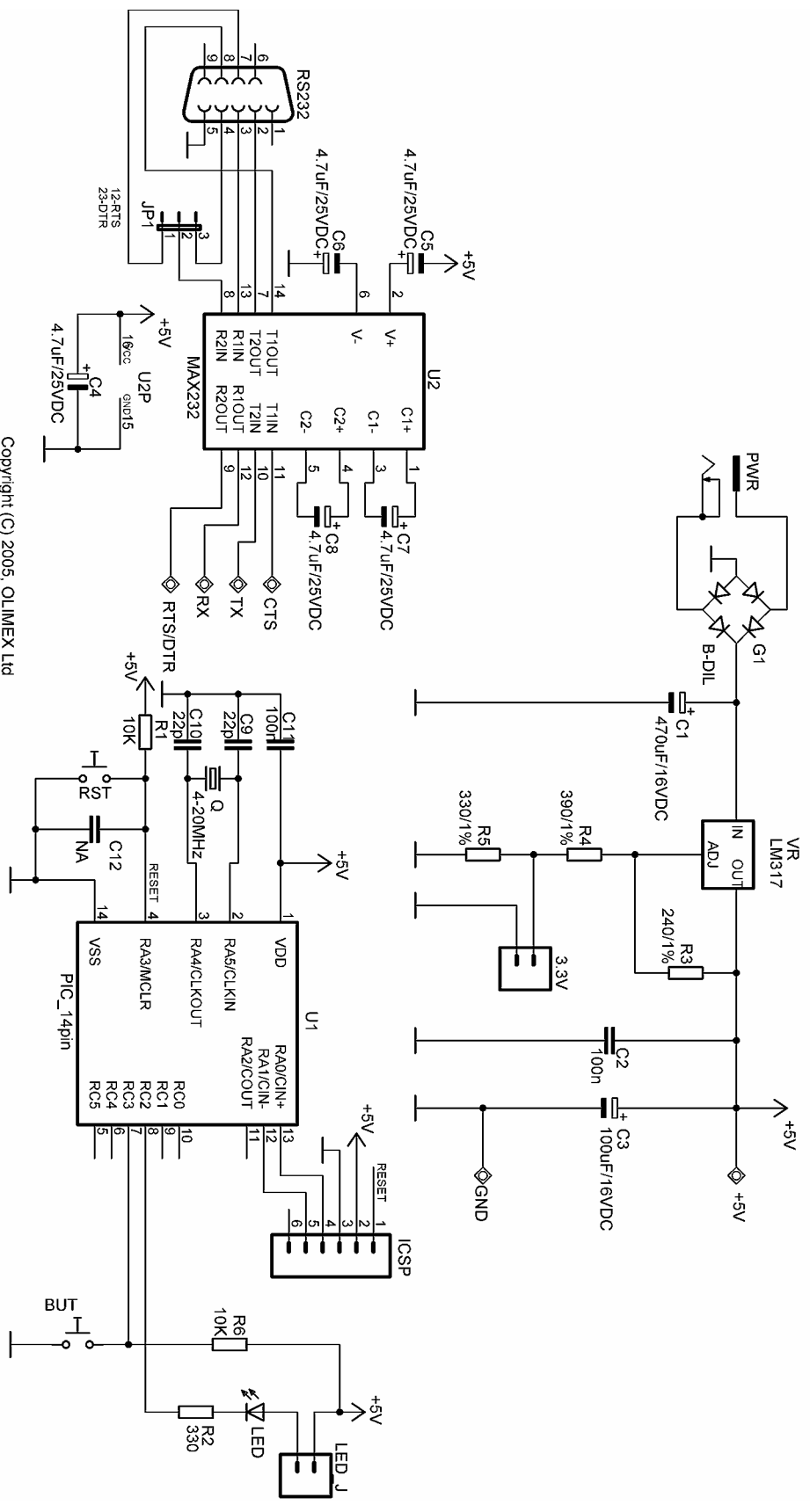
SOFTWARE:

DEMO1:   PIC16F688-I/P   BLINK LED

This is demo code which blinks the LED on PIC-P14 board.

Note:
LED jumper should be closed!

```
//PROGRAMM FOR BLINKING LED
//
// PROCESSOR : PIC16F688
// CLOCK        : 20MHz, EXTERNAL

#include        <pic.h>

__CONFIG(WDTDIS & MCLREN & BORDIS & HS & PWRTEN & FCMDIS & IESODIS) ;

//Just simple delay
void Delay(unsigned long cntr) {
                while (--cntr != 0);
}

// main function
void main( void ) {

                OPTION      = 0x80;       // GPIO pull-ups are disabled
                INTCON      = 0x0;        // Disable inerupt
                WPUA  = 0x0;              // pull-up disable
                ANSEL       = 0x0;        // port as Digital IO
                CMCON0      = 0x07;       // port as Digital IO
                TRISC2      = 0;          // Led pin as output

                // loop forever - echo
                while(1) {

                    // Toggle led
                    PORTC ^= 0x04;
                    // Simple delay
                    Delay(50000);
                }
}
```

DEMO2:   PIC16F688-I/P   BUTTON read
This is demo code which show how to read BUTTON status

DEMO3:   PIC16F688-I/P   RS232 send / receive routines

This is demo code which show how to use the USART to send and receive characters from host PC via RS232 cable.

The MAX232 by default is not connected to PIC on the board so you have to solder two wires – one from RC4 (pin #6) of PIC16F688 Tx signal to MAX232 Tx labeled pin, and second from RC5 (pin #5) of PIC16F688 Rx signal to MAX232 Rx labeled pin.

You must program the HEX code to PIC16F688 and run the code.

If you programmed the PIC correctly and wired the UART, when you open Hyperterminal on your host PC computer with 9600 bps, 8 data bit, 1 stop bit, No flow control and apply power supply to the PIC-P14 every character

you type on the hyperterminal will be printed back with "*" i.e. if you type "abc" you will receive "a*b*c*".

```c
//PROGRAMM FOR rs232 communication
//
// PROCESSOR : PIC16F688
// CLOCK     : 20MHz, EXTERNAL
// SPEED     : 9600 bps(1bit=104us)

#include        <pic.h>

__CONFIG(WDTDIS & MCLREN & BORDIS & HS & PWRTEN & FCMDIS & IESODIS) ;

unsigned char ch;

void InitUsart(void) {

                // TX Pin - output
                TRISC4 = 0;

                // RX Pin - input
                TRISC5 = 1;

                // RX Setting, 8bit, enable receive,
                RCSTA = 0x90;

                // TX Setting, 8bit, Asinchronius mode, High speed
                TXSTA = 0x24;

                // Baudcontrol - disable auto mode, 8bit baudrate
                BAUDCTL = 0x00;

                // Set Baudrade - 9600 (from datasheet baudrade table)
                SPBRG = 129;
}

void  WriteByte(unsigned char byte) {

                // wait until register is empty
                while(!TXIF);

                // transmite byte
                TXREG = byte;
}

unsigned char ReadByte(void) {

                // wait to receive character
                while(!RCIF);

                // return received character
                return RCREG;
}


// main function
void main( void ) {

                // Init Interface
                InitUsart();

                // loop forever - echo
                while(1) {
                    ch = ReadByte();        //wait to receive character
                    WriteByte(ch);          //send it back followed by "*"
                    WriteByte('*');
                }
}
```

ORDER CODE:

PIC-P14 – assembled and tested (no kit, no soldering required)

PIC-P14/PCB – blank PCBs only

Revision history:

REV.A   - create

REV.B   - add software demo codes