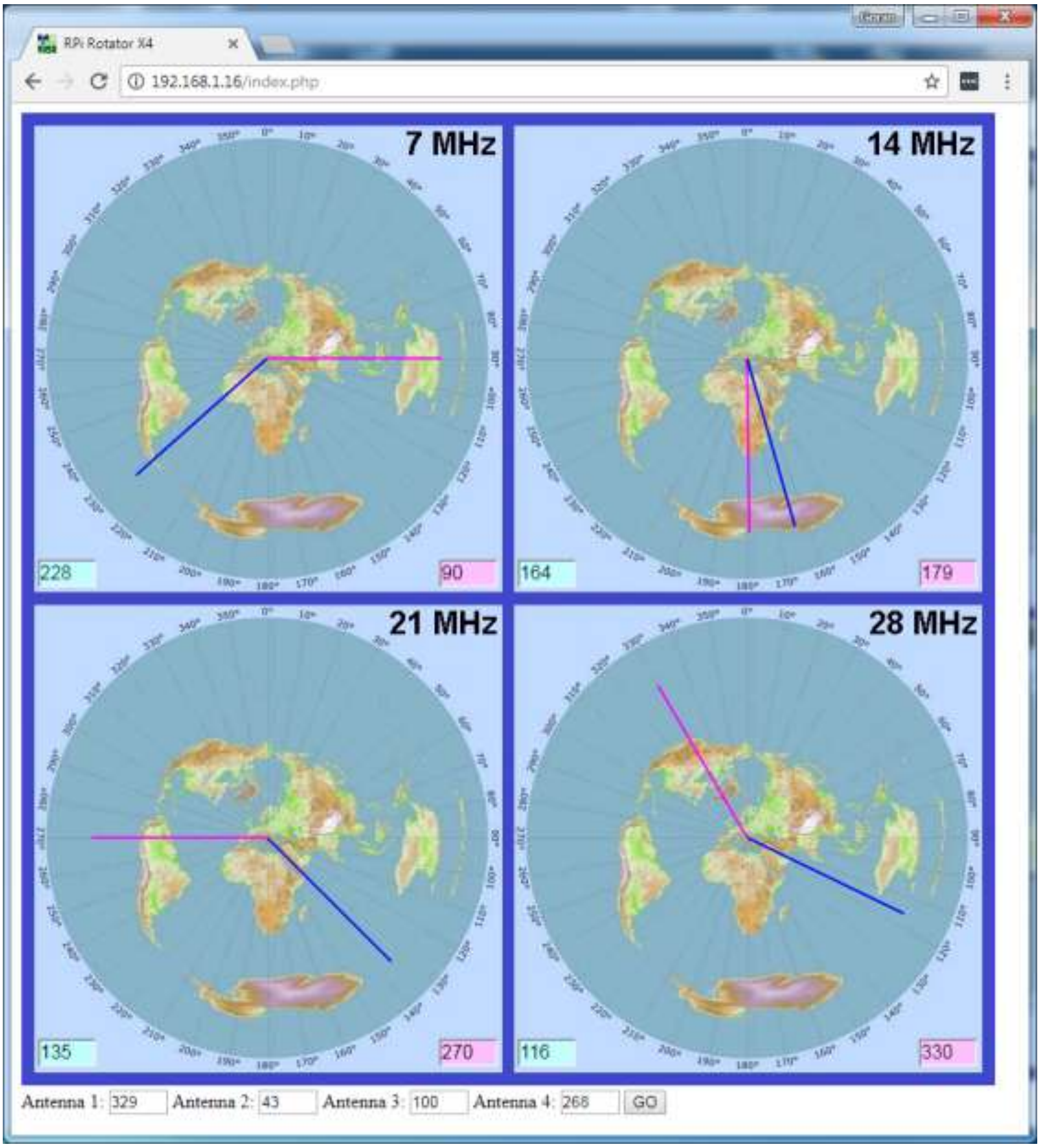
RPi Rotator X4

by Goran Stankovic dip.ing.el. – YT2FSG – Date: 04.10.2016.

email: goranstank@gmail.com



RPi Rotator X4

192.168.1.16/index.php

7 MHz

228 90

14 MHz

164 179

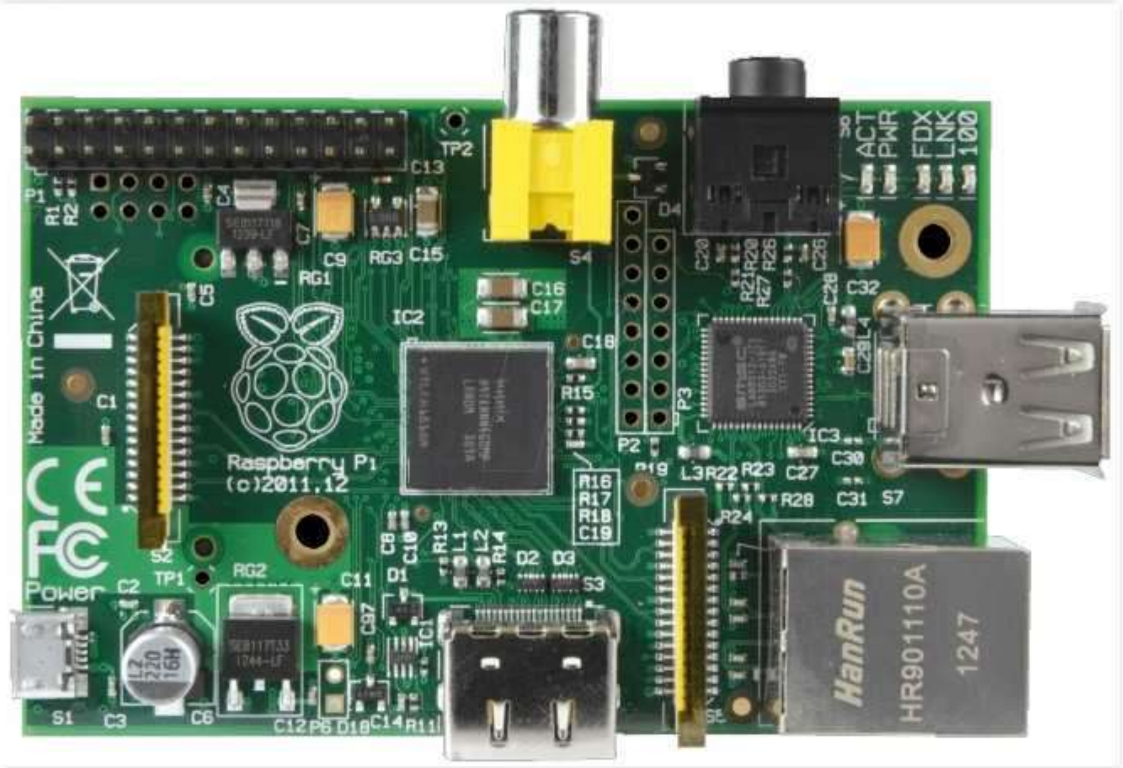
21 MHz

135 270

28 MHz

116 330

Antenna 1: 329 Antenna 2: 43 Antenna 3: 100 Antenna 4: 268 GO



LINUX – najčešće komande

`$ sudo ime_kom`	substitute user do (omogućuje izvršavanje komandi kao drugi korisnik)
`$ pwd`	prikaži ime radnog direktorijuma
`$ cd ime_dir`	promeni direktorijum
`$ ls`	lista fajlova u direktorijumu
`$ ls -altr`	lista fajlova u direktorijumu, poredjaj po vremenu formiranja fajla
`$ mkdir ime_dir`	kreiraj novi direktorijum
`$ rmdir ime_dir`	izbriši direktorijum
`$ rm ime`	izbriši fajl
`$ rm *.txt`	izbriši fajlove sa ekstenzijom txt
`$ touch ime`	kreiraj tekst fajl u tekućem direktorijumu
`$ mv ime1 ime2`	promeni ime fajla
`$ cp ime /ime_dir`	kopiraj fajl u direktorijum
`$ less ime`	aplikacija za pregledanja fajlova (prekid pritiskom na Q)
`$ nano ime`	tekst editor
`$ ./ime_prg`	izvrši program
`$ exit`	izlazak iz shell-a ako smo logovani kao root ili neki drugi korisnik
`$ man ime_kom`	upustvo za komande i ostalo
`$ df`	prikaži prostor na disku
`$ free`	prikaži prostor RAM memorije
`$ ps aux`	lista svih procesa
`$ top`	zauzeće procesora
`$ tar cvf ime.tar /ime_dir`	kompresija fajlova sa imenom direktorija
`$ tar xvf ime.tarr`	dekompresija fajlova
`$ zip ime.zip *.*`	kompresija svih fajlova u direktorijumu
`$ unzip –La ime.zip`	dekompresija fajlova
`$ chown ime`	promena vlasništva fajla ili direktorijuma
`$ chmod ime`	promena dozvola na fajlu ili folderu, prvi broj predstavlja kreatora, drugi grupu i tereći bilo koga. (read = 4, write = 2, exec = 1)
`$ chmod 000`	nije dozvoljeno ništa
`$ chmod 644`	najčešće dozvole
`$ chmod 754`	korisnik – može sve, grupa – da čita i izvršava, i bilo ko – može samo da čita
`$ chmod 755`	korisnik – može sve, grupa – da čita i izvršava, i bilo ko – da čita i izvršava
`$ id`	identifikacija korisnika
`$ date`	prikaži sistemsko vreme

nano - tekst editor komande:

Ctrl+G = HELP
Ctrl+K = CAT LINE
Alt+6 = COPY LINE
Ctrl+U = PASTE LINE

Ctrl+O = SAVE FILE
Ctrl+X = QUIT
Ctrl+C = Cursor position
Ctrl+W = Search text
Alt+W = Replace
Alt+R = Search and replace

Instalacija operativnog sistema

Nakon instaliranja novog operativnog sistema na vašu SD karticu Raspberry Pi, možda ćete primetiti da na kartici ima puno 'nedostajućeg' disk prostora ...
Širenjem root particije kartice možete vratiti taj 'nedostajući' prostor!
Ovo je najlakši način da to učinite:
Otvorite Raspberry Pi's konfiguracioni ekran (u terminal window-su), pomoću komande:

```
$ sudo raspi-config
```

```
$ df -h                prikaži prostor na disku
$ free -m              prikaži prostor RAM memorije u MB
```

Instalacija Apache servera i PHP:

```
 $ sudo apt-get update
 $ sudo apt-get upgrade
 $ sudo apt-get install apache2
 $ sudo apt-get install php5
```

Instalacija FTP servera:

```
$ sudo apt-get update
$ sudo apt-get install vsftpd
```

U fajlu `/etc/vsftpd.config` dodati ili skinuti (pomoću #)

```
$ sudo nano /etc/vsftpd.config
.
.
.
anonymous_enable=NO
local_enable=YES
write_enable=YES
local_umask=022
chroot_local_user=YES
user_sub_token=$USER
local_root=/home/$USER/ftp

$ mkdir /home/pi/ftp
$ mkdir /home/pi/ftp/files
$ chmod a-w /home/pi/ftp

$ sudo service vsftpd restart
```

Instalacija Midnight Commandera:

```
 $ sudo apt-get install mc
```

Configuracija I2C

```
$ sudo apt-get install python-smbus
$ sudo apt-get install i2c-tools
```

u File `/etc/modprobe.d/raspi-blacklist_config` dodati #:

```
$ sudo nano /etc/modprobe.d/raspi-blacklist_config
# i2c_bcm2708
```

u File `/etc/modules` dodati:

```
$sudo nano /etc/modules
i2c-dev
i2c-bcm2708
```

Pokrenite ponovo sistem

```
$ sudo reboot
```

Posle podizanaja sistema testirajte

```
$ sudo i2cdetect -y 1
```

Formiranje RAM DISK

```
$ sudo mkdir /var/tmp
```

u File `/etc/fstab` dodati:

```
$ sudo nano /etc/fstab
.
.
.
tmpfs /var/tmp tmpfs nodev,nosuid,size=1M 0 0

$ sudo mount -a
$ df
```

Instalacija dodatnog modula, sata realnog vremena, RTC DS3231

u File `/etc/ rc.local` dodati:

```
$ sudo nano /etc/rc.local
#!/bin/sh -e
.
.
.
echo ds3231 0x68 > /sys/class/i2c-adapter/i2c-1/new-device
sudo hwclock -s
exit 0
```

Pokrenite ponovo sistem

```
$ sudo reboot
```

Posle podizanaja sistema testirajte

```
$ sudo hwclock
```

Instalacija software RPi Rotator X4

Kompajlirajte softver pomoću gcc:

```
$ gcc -o rotator4 -l rt rotator4.c -l bcm2835
```

Testirajte softver pomoću:

```
$ ./rotator4
```

U fajlu /etc/rc.local:

```
$ sudo nano /etc/rc.local
```

Dodati liniju:

```
sudo /home/pi/rotator4
```

Raspakujte fajlove yu5r_rotator`.tar` u directorijum `/var/www`

```
$ tar xvf rpi_rotator_x4.tar

$ ls /var/www/
index.php              styles.css     yu5r_360_4.jpg        yu5r.ico
pos_set                position       set
```

Razmena podataka izmedju `/var/www/index.php` i programa `/home/rpi/rotator4` je pomoću fajlova:

Podaci o trenutnom položaju antena nalaze se na RAM-DISK-u (rotator4 upisuje, PHP čita):
File: `/var/tmp/position`

```
228,164,135,116
```

Podaci o zadatom položaju antena nalaze se na FLASH-DISK-u (PHP upisuje, rotator4 čita):
File: `/var/www/pos_set`

```
90,179,270,330
```

Podatak o komandi **GO** za pokretanje motora rotatora nalazi se na FLASH-DISK-u (PHP upisuje(1), rotator4 čita - upisuje(0)):
File: `/var/www/set`

```
1
```

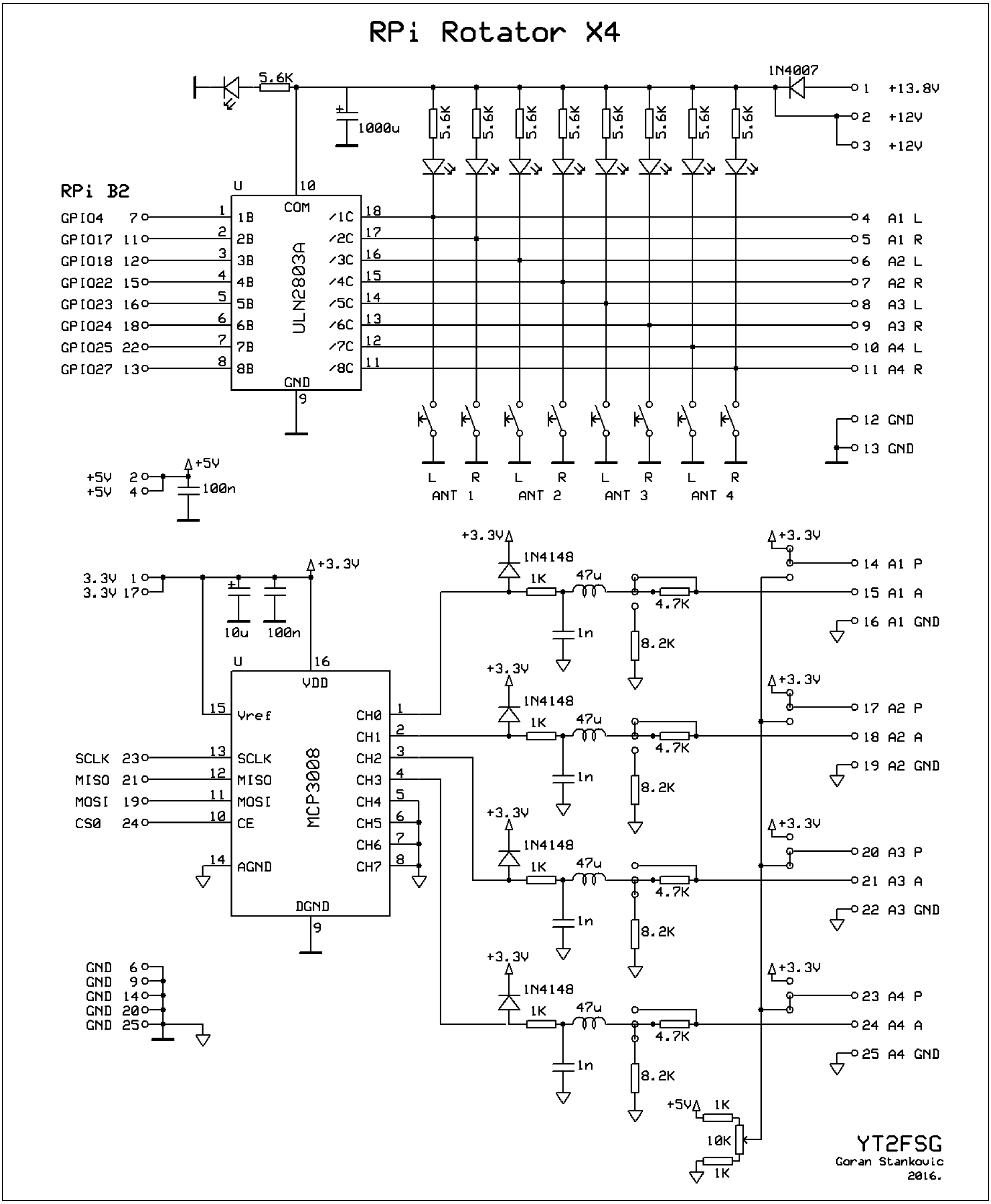
Parametre koristi program `/home/rpi/rotator4` nalaze se na FLASH-DISK-u:

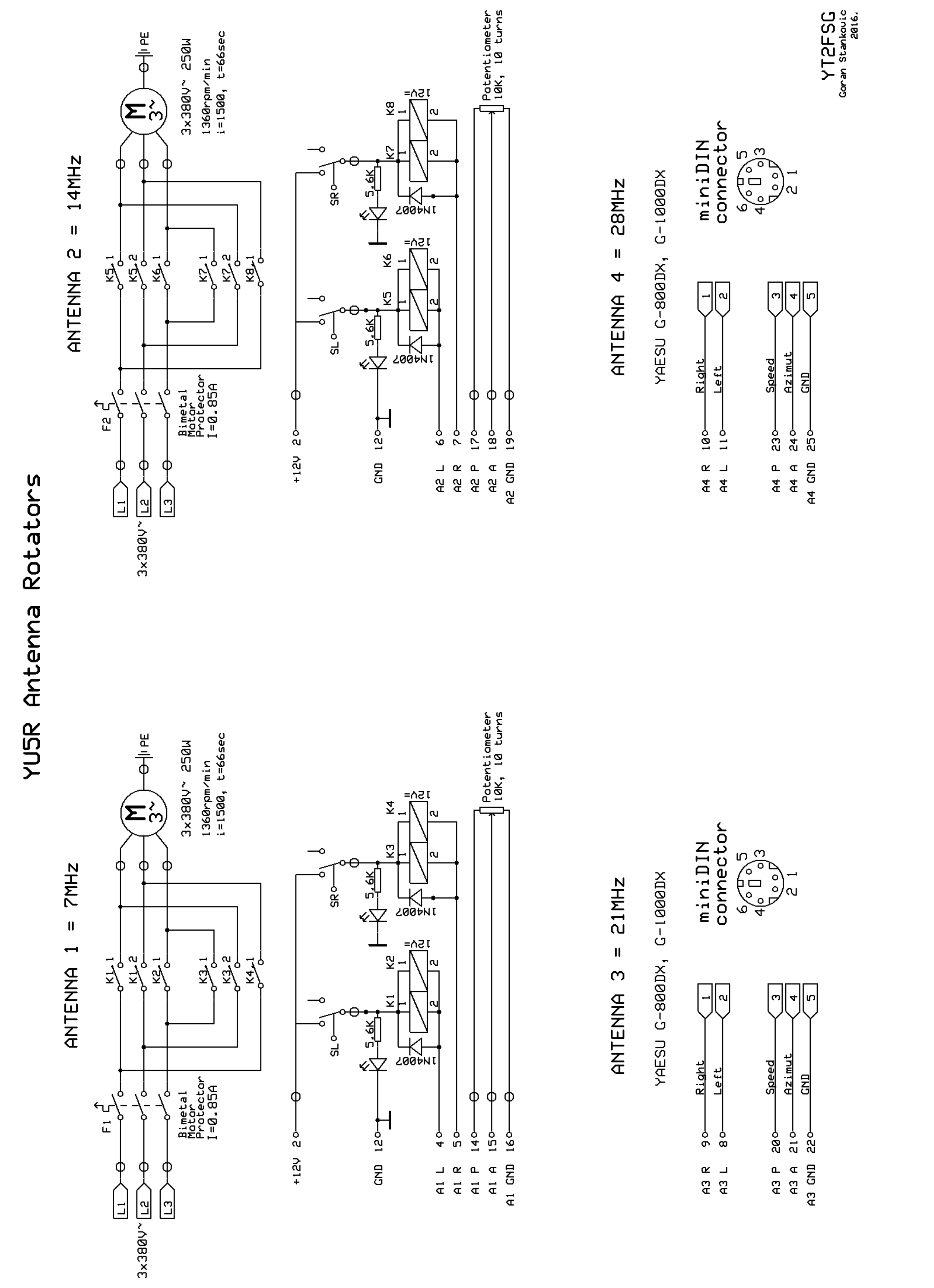
Parametri za kalibraciju pokazivanja trenutnog položaja antena 0° i 360° su u binarnim jedinicama, vrednost koju čita ADC MCP3008 : pmin1, pmax1, pmin2, pmax2, pmin3, pmax3, pmin4, pmax4;
File: `/home/rpi/calibration`

```
50,1000,50,1000,50,1000,50,1000
```

Parametri maksimalno vreme rada motora rotatora posle komande **GO** su: time_max1, time_max2, time_max3, time_max4;
File: `/home/rpi/time_max`

```
80,80,120,180
```


RPi Rotator X4
RPi B2
GPIO4 7
GPIO17 11
GPIO18 12
GPIO22 15
GPIO23 16
GPIO24 18
GPIO25 22
GPIO27 13
ULN2803A
COM
GND
1B 2B 3B 4B 5B 6B 7B 8B
1C 2C 3C 4C 5C 6C 7C 8C
5.6K
1000u
1N4007
1 +13.8V
2 +12V
3 +12V
4 A1 L
5 A1 R
6 A2 L
7 A2 R
8 A3 L
9 A3 R
10 A4 L
11 A4 R
12 GND
13 GND
L R L R L R L R
ANT 1 ANT 2 ANT 3 ANT 4
+5V 2
+5V 4
+5V
100n
3.3V 1
3.3V 17
+3.3V
10u
100n
MCP3008
VDD
Vref
SCLK
MISO
MOSI
CE
AGND
DGND
CH0 CH1 CH2 CH3 CH4 CH5 CH6 CH7
SCLK 23
MISO 21
MOSI 19
CS0 24
GND 6
GND 9
GND 14
GND 20
GND 25
1N4148
1K
47u
1n
4.7K
8.2K
14 A1 P
15 A1 A
16 A1 GND
17 A2 P
18 A2 A
19 A2 GND
20 A3 P
21 A3 A
22 A3 GND
23 A4 P
24 A4 A
25 A4 GND
+5V
10K
YT2FSG
Goran Stankovic
2016.


YU5R Antenna Rotators
ANTENNA 1 = 7MHz
ANTENNA 2 = 14MHz
ANTENNA 3 = 21MHz
ANTENNA 4 = 28MHz
YAESU G-800DX, G-1000DX
miniDIN connector
1 2 3 4 5 6
Right
Left
Speed
Azimut
GND
A3 R 9
A3 L 8
A3 P 20
A3 A 21
A3 GND 22
A4 R 10
A4 L 11
A4 P 23
A4 A 24
A4 GND 25
3x380V~
L1
L2
L3
F1
F2
Bimetal Motor Protector I=0.85A
K1.1 K1.2 K2.1 K3.1 K3.2 K4.1
K5.1 K5.2 K6.1 K7.1 K7.2 K8.1
PE
M 3~
3x380V~ 250W
1360rpm/min
i=1500, t=66sec
+12V 2
GND 12
SL
SR
5.6K
1N4007
K1 K2 K3 K4 K5 K6 K7 K8
12V=
A1 L 4
A1 R 5
A1 P 14
A1 A 15
A1 GND 16
A2 L 6
A2 R 7
A2 P 17
A2 A 18
A2 GND 19
Potentiometer 10K, 10 turns
YT2FSG
Goran Stankovic 2016.

File: /var/www/index.php

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<meta name="author" content="Goran Stankovic, YT2FSG">
<title>RPi Rotator 4</title>
<link rel="icon" href="yu5r.ico" type="image/ico" />

<script language="JavaScript">
function getDetails(obj)
{
  var clickX = window.event.x-obj.offsetLeft;
  var clickY = window.event.y-obj.offsetTop;

 if (clickX > 10 && clickX < 410 && clickY > 10 && clickY < 410) {
  clickX=210-clickX;
  clickY=clickY-210;
  var azi = clickY / clickX;
  azi = Math.atan(azi);
  azi=parseInt(90-57.29*azi);
  if (clickX >= 0)
  {
    azi=180+azi;
  }
  document.getElementById("p_az1").value = azi;

 } else if (clickX > 420 && clickX < 820 && clickY > 10 && clickY < 410) {
  clickX=620-clickX;
  clickY=clickY-210;
  var azi = clickY / clickX;
  azi = Math.atan(azi);
  azi=parseInt(90-57.29*azi);
  if (clickX >= 0)
  {
    azi=180+azi;
  }
  document.getElementById("p_az2").value = azi;

 } else if (clickX > 10 && clickX < 410 && clickY > 420 && clickY < 820) {
  clickX=210-clickX;
  clickY=clickY-620;
  var azi = clickY / clickX;
  azi = Math.atan(azi);
  azi=parseInt(90-57.29*azi);
  if (clickX >= 0)
  {
    azi=180+azi;
  }
  document.getElementById("p_az3").value = azi;

 } else if (clickX > 420 && clickX < 820 && clickY > 420 && clickY < 820) {
  clickX=620-clickX;
  clickY=clickY-620;
  var azi = clickY / clickX;
  azi = Math.atan(azi);
  azi=parseInt(90-57.29*azi);
  if (clickX >= 0)
  {
    azi=180+azi;
  }
  document.getElementById("p_az4").value = azi;
 }
}

function draw()
{
  var canvas = document.getElementById('cav1');

  if (canvas.getContext)
  {
    var ctx = canvas.getContext('2d');
    ctx.drawImage(mapa_4, 0, 0);

    ctx.beginPath();
    ctx.moveTo(210, 210);
    ctx.lineTo(a_x1, a_y1);
    ctx.strokeStyle = '#FF00FF';
    ctx.lineWidth = 2;
    ctx.stroke();
    ctx.beginPath();
    ctx.moveTo(210, 210);
    ctx.lineTo(p_x1, p_y1);
    ctx.strokeStyle = '#0000FF';
    ctx.lineWidth = 2;
    ctx.stroke();

    ctx.beginPath();
    ctx.moveTo(620, 210);
    ctx.lineTo(a_x2, a_y2);
    ctx.strokeStyle = '#FF00FF';
    ctx.lineWidth = 2;
    ctx.stroke();
    ctx.beginPath();
    ctx.moveTo(620, 210);
    ctx.lineTo(p_x2, p_y2);
    ctx.strokeStyle = '#0000FF';
    ctx.lineWidth = 2;
    ctx.stroke();

    ctx.beginPath();
    ctx.moveTo(210, 620);
    ctx.lineTo(a_x3, a_y3);
    ctx.strokeStyle = '#FF00FF';
    ctx.lineWidth = 2;
    ctx.stroke();
    ctx.beginPath();
    ctx.moveTo(210, 620);
    ctx.lineTo(p_x3, p_y3);
    ctx.strokeStyle = '#0000FF';
    ctx.lineWidth = 2;
    ctx.stroke();

    ctx.beginPath();
    ctx.moveTo(620, 620);
    ctx.lineTo(a_x4, a_y4);
    ctx.strokeStyle = '#FF00FF';
    ctx.lineWidth = 2;
    ctx.stroke();
    ctx.beginPath();
    ctx.moveTo(620, 620);
    ctx.lineTo(p_x4, p_y4);
    ctx.strokeStyle = '#0000FF';
    ctx.lineWidth = 2;
    ctx.stroke();

  }
  else
  {
    // make the element with the no-canvas message visible
    document.getElementById('no_cav').style.display = 'block';
  }
}
</script>
</head>


<body onload="draw()">

<canvas width="832" height="832" id="cav1" onclick="getDetails(this)">
  <img id="mapa_4" src="yu5r_360_4.jpg" >
</canvas>
```

```
<p id="no_cav" style="display:none;">You need a Canvas-compatible browser to view canvas area.</p>

<?php
echo '<link rel="stylesheet" href="styles.css">';

//echo "<br>";

$filename = "/var/tmp/position";
//$filename = "position";
$handle = fopen($filename, "r");
$azim_f = fread($handle, filesize($filename));
fclose($handle);

$azim_f1=$azim_f;
$p1 = substr($azim_f1, 0, strpos($azim_f1, ',') );
$azim_f1 = substr($azim_f1, strpos($azim_f1, ',')+1 );
$p2 = substr($azim_f1, 0, strpos($azim_f1, ',') );
$azim_f1 = substr($azim_f1, strpos($azim_f1, ',')+1 );
$p3 = substr($azim_f1, 0, strpos($azim_f1, ',') );
$p4 = substr($azim_f1, strpos($azim_f1, ',')+1 );

$filename = "pos_set";
$handle = fopen($filename, "r");
$azim_f = fread($handle, filesize($filename));
fclose($handle);

$azim_f1=$azim_f;
$a1 = substr($azim_f1, 0, strpos($azim_f1, ',') );
$azim_f1 = substr($azim_f1, strpos($azim_f1, ',')+1 );
$a2 = substr($azim_f1, 0, strpos($azim_f1, ',') );
$azim_f1 = substr($azim_f1, strpos($azim_f1, ',')+1 );
$a3 = substr($azim_f1, 0, strpos($azim_f1, ',') );
$a4 = substr($azim_f1, strpos($azim_f1, ',')+1 );

//print("$p1 $p2 $p3 $p4");

echo '<input class="p1" type="text" value="'.$p1.'" name="p1" maxlength="4">';
echo '<input class="p2" type="text" value="'.$p2.'" name="p2" maxlength="4">';
echo '<input class="p3" type="text" value="'.$p3.'" name="p3" maxlength="4">';
echo '<input class="p4" type="text" value="'.$p4.'" name="p4" maxlength="4">';
echo '<input class="a1" type="text" value="'.$a1.'" name="a1" maxlength="4">';
echo '<input class="a2" type="text" value="'.$a2.'" name="a2" maxlength="4">';
echo '<input class="a3" type="text" value="'.$a3.'" name="a3" maxlength="4">';
echo '<input class="a4" type="text" value="'.$a4.'" name="a4" maxlength="4">';

$az1=(180-$a1)*0.01744;
$az_x1=150*(sin($az1))+210;
$az_y1=150*(cos($az1))+210;
$az_x1=(int)$az_x1;
$az_y1=(int)$az_y1;

$pz1=(180-$p1)*0.01744;
$pz_x1=150*(sin($pz1))+210;
$pz_y1=150*(cos($pz1))+210;
$pz_x1=(int)$pz_x1;
$pz_y1=(int)$pz_y1;

$az2=(180-$a2)*0.01744;
$az_x2=150*(sin($az2))+620;
$az_y2=150*(cos($az2))+210;
$az_x2=(int)$az_x2;
$az_y2=(int)$az_y2;

$pz2=(180-$p2)*0.01744;
$pz_x2=150*(sin($pz2))+620;
$pz_y2=150*(cos($pz2))+210;
$pz_x2=(int)$pz_x2;
$pz_y2=(int)$pz_y2;

$az3=(180-$a3)*0.01744;
$az_x3=150*(sin($az3))+210;
```

```
$az_y3=150*(cos($az3))+620;
$az_x3=(int)$az_x3;
$az_y3=(int)$az_y3;

$pz3=(180-$p3)*0.01744;
$pz_x3=150*(sin($pz3))+210;
$pz_y3=150*(cos($pz3))+620;
$pz_x3=(int)$pz_x3;
$pz_y3=(int)$pz_y3;

$az4=(180-$a4)*0.01744;
$az_x4=150*(sin($az4))+620;
$az_y4=150*(cos($az4))+620;
$az_x4=(int)$az_x4;
$az_y4=(int)$az_y4;

$pz4=(180-$p4)*0.01744;
$pz_x4=150*(sin($pz4))+620;
$pz_y4=150*(cos($pz4))+620;
$pz_x4=(int)$pz_x4;
$pz_y4=(int)$pz_y4;

echo '
<script type="text/javascript">
// <![CDATA[
    a_x1 = "' . $az_x1 . '";
    a_y1 = "' . $az_y1 . '";
    p_x1 = "' . $pz_x1 . '";
    p_y1 = "' . $pz_y1 . '";
    a_x2 = "' . $az_x2 . '";
    a_y2 = "' . $az_y2 . '";
    p_x2 = "' . $pz_x2 . '";
    p_y2 = "' . $pz_y2 . '";
    a_x3 = "' . $az_x3 . '";
    a_y3 = "' . $az_y3 . '";
    p_x3 = "' . $pz_x3 . '";
    p_y3 = "' . $pz_y3 . '";
    a_x4 = "' . $az_x4 . '";
    a_y4 = "' . $az_y4 . '";
    p_x4 = "' . $pz_x4 . '";
    p_y4 = "' . $pz_y4 . '";
// ]]>
</script>';

echo "<br>";



if(!isset($_REQUEST['p_az1']))
{
?>
    <form method="get" name="input">
      Antenna 1: <input id="p_az1" name="p_az1" value="" style="width: 50px;" maxlength="3">
      Antenna 2: <input id="p_az2" name="p_az2" value="" style="width: 50px;" maxlength="3">
      Antenna 3: <input id="p_az3" name="p_az3" value="" style="width: 50px;" maxlength="3">
      Antenna 4: <input id="p_az4" name="p_az4" value="" style="width: 50px;" maxlength="3">
      <input type="submit" value="GO" >
    </form>
<?php
    exit();
}

$azim_1 = $_REQUEST['p_az1'];
$azim_2 = $_REQUEST['p_az2'];
$azim_3 = $_REQUEST['p_az3'];
$azim_4 = $_REQUEST['p_az4'];

if (strlen($azim_1)<1) {  $azim_1=$p1;}
if (strlen($azim_2)<1) {  $azim_2=$p2;}
if (strlen($azim_3)<1) {  $azim_3=$p3;}
if (strlen($azim_4)<1) {  $azim_4=$p4;}
```

```
print("Antenna 1 = $azim_1, Antenna 2 = $azim_2, Antenna 3 = $azim_3, Antenna 4 = $azim_4");
echo "<br>";

$filename = "pos_set";
$handle = fopen($filename, "w+");
fwrite($handle, $azim_1);
fwrite($handle, ",");
fwrite($handle, $azim_2);
fwrite($handle, ",");
fwrite($handle, $azim_3);
fwrite($handle, ",");
fwrite($handle, $azim_4);
fclose($handle);

$filename = "set";
$handle = fopen($filename, "w+");
fwrite($handle, '1');
fclose($handle);

echo "<br>";
?>
<meta http-equiv="refresh" content="1;url=index.php">
</body>
</html>
```

File: /home/rpi/rotator4.c

```
// RPi Rotator X4 with MCP3008
// 25.09.2016. Author: Goran Stankovic dipl.ing.el. (goranstank@gmail.com)
//
// gcc -o rotator -l rt rotator.c -l bcm2835
// sudo ./rotator

#include <bcm2835.h>
#include <stdio.h>
#include <stdlib.h>


int main(int argc, char **argv)
{

    FILE * fp;

    if (!bcm2835_init())
        return 1;

    bcm2835_spi_begin();
    bcm2835_spi_setBitOrder(BCM2835_SPI_BIT_ORDER_MSBFIRST);
    bcm2835_spi_setDataMode(BCM2835_SPI_MODE0);
    bcm2835_spi_setClockDivider(BCM2835_SPI_CLOCK_DIVIDER_256);     // 250MHz/256=976.5kHz
    bcm2835_spi_chipSelect(BCM2835_SPI_CS0);                        // Slave on CS0
    bcm2835_spi_setChipSelectPolarity(BCM2835_SPI_CS0, LOW);

    bcm2835_gpio_fsel( 4, BCM2835_GPIO_FSEL_OUTP);
    bcm2835_gpio_fsel(17, BCM2835_GPIO_FSEL_OUTP);
    bcm2835_gpio_fsel(18, BCM2835_GPIO_FSEL_OUTP);
    bcm2835_gpio_fsel(22, BCM2835_GPIO_FSEL_OUTP);
    bcm2835_gpio_fsel(23, BCM2835_GPIO_FSEL_OUTP);
    bcm2835_gpio_fsel(24, BCM2835_GPIO_FSEL_OUTP);
    bcm2835_gpio_fsel(25, BCM2835_GPIO_FSEL_OUTP);
    bcm2835_gpio_fsel(27, BCM2835_GPIO_FSEL_OUTP);


    char buf[] = { 0x01, 0x80, 0x00 }; // Data to send
    int adc1, adc2, adc3, adc4;
    int p1, p2, p3, p4;
    int set=1;        // Start pozicioniranja
    int delta=2;      // min ugao za koji rotator ne reaguje
    int out1=0, out2=0, out3=0, out4=0;
    int ps1=100, ps2=200, ps3=300, ps4=340;
    int time1=0, time2=0, time3=0, time4=0;
    int time_max1=60*4, time_max2=60*4, time_max3=60*4, time_max4=60*4;
    int pmin1=0, pmax1=1023, pmin2=0, pmax2=1023, pmin3=0, pmax3=1023, pmin4=0, pmax4=1023;
    int op1, op2, op3, op4;

    char c;
    char buffer[100];
    long length;
    int i=0, j=0;
    int buffer2[16];
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
    printf("*** RPi Rotator for 4 antennas system ***  YT2FSG  ***\n");
    printf("by: Goran Stankovic dipl.ing.el. (goranstank@gmail.com)\n\n");
       printf("Parameters:   Ant1 Ant2 Ant3 Ant4\n");
// Ucitavanje parametara iz file 'time_max'
    if ((fp = fopen("time_max","r")) == NULL)
    {
       printf("file 'time_max' not found! \n");
    }
    else
    {
       while(1)
       {
         c = fgetc(fp);
         if (feof(fp)) { break; }
         if (c==',') { buffer[i] = 0; i = 0;  buffer2[j] = atoi(buffer); j++; }
         else { buffer[i] = c; i++; }
       }
         buffer[i] = 0;
         buffer2[j] = atoi(buffer);

               printf("time_max (sec): %3d %3d %3d %3d \n", buffer2[0], buffer2[1], buffer2[2], buffer2[3]);

        time_max1 = buffer2[0] * 4;
        time_max2 = buffer2[1] * 4;
        time_max3 = buffer2[2] * 4;
        time_max4 = buffer2[3] * 4;
    }
    fclose(fp);
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// Ucitavanje parametara iz file 'calibration'
    i=0; j=0;
    if ((fp = fopen("calibration","r")) == NULL)
    {
       printf("file 'calibration' not found! \n");
    }
    else
    {
       while(1)
       {
         c = fgetc(fp);
         if (feof(fp)) { break; }
         if (c==',') { buffer[i] = 0; i = 0;  buffer2[j] = atoi(buffer); j++; }
         else { buffer[i] = c; i++; }
       }
         buffer[i] = 0;
         buffer2[j] = atoi(buffer);

               printf("cal_min  (   ): %3d %3d %3d %3d \n", buffer2[0], buffer2[2], buffer2[4], buffer2[6]);
               printf("cal_max  (   ): %3d %3d %3d %3d \n", buffer2[1], buffer2[3], buffer2[5], buffer2[7]);

               pmin1 = buffer2[0]; pmax1 = buffer2[1];
               pmin2 = buffer2[2]; pmax2 = buffer2[3];
               pmin3 = buffer2[4]; pmax3 = buffer2[5];
               pmin4 = buffer2[6]; pmax4 = buffer2[7];
     }
    fclose(fp);

//    exit(0);
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

```
  while (1)
  {
    // Citanje pozicije antena - potenciometri (MCP3008)
    // CH0
    buf[0] = 0x01; buf[1] = 0x80; buf[2] = 0x00;
    bcm2835_spi_transfern(buf, sizeof(buf));
    adc1 = buf[1] & 0b00000011;
    adc1 = (adc1 << 8) | buf[2];

    // CH1
    buf[0] = 0x01; buf[1] = 0x90; buf[2] = 0x00;
    bcm2835_spi_transfern(buf, sizeof(buf));
    adc2 = buf[1] & 0b00000011;
    adc2 = (adc2 << 8) | buf[2];

    // CH2
    buf[0] = 0x01; buf[1] = 0xA0; buf[2] = 0x00;
    bcm2835_spi_transfern(buf, sizeof(buf));
    adc3 = buf[1] & 0b00000011;
    adc3 = (adc3 << 8) | buf[2];

    // CH3
    buf[0] = 0x01; buf[1] = 0xB0; buf[2] = 0x00;
    bcm2835_spi_transfern(buf, sizeof(buf));
    adc4 = buf[1] & 0b00000011;
    adc4 = (adc4 << 8) | buf[2];

// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// Calibration
    op1 = pmax1 - pmin1;
       if (op1 < 1 ) { p1 = adc1 * 360 / 1023; }
       else { p1 = (adc1 - pmin1) * 360 / op1; }
    op2 = pmax2 - pmin2;
       if (op2 < 1 ) { p2 = adc2 * 360 / 1023; }
       else { p2 = (adc2 - pmin2) * 360 / op2; }
    op3 = pmax3 - pmin3;
       if (op3 < 1 ) { p3 = adc3 * 360 / 1023; }
       else { p3 = (adc3 - pmin3) * 360 / op3; }
    op4 = pmax4 - pmin4;
       if (op4 < 1 ) { p4 = adc4 * 360 / 1023; }
       else { p4 = (adc4 - pmin4) * 360 / op4; }

//    printf("Position: %d=%d %d=%d %d=%d %d=%d \n", adc1, p1, adc2, p2, adc3, p3, adc4, p4);

// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// Ucitavanje parametara iz FLASH-DISK file 'pos_set'
    i=0; j=0;
    if ((fp = fopen("/var/www/pos_set","r")) == NULL)
    {
       printf("file '/var/www/pos_set' not found! \n");
    }
    else
    {
       while(1)
       {
         c = fgetc(fp);
         if (feof(fp)) { break; }
         if (c==',') { buffer[i] = 0; i = 0;  buffer2[j] = atoi(buffer); j++; }
         else { buffer[i] = c; i++; }
       }
         buffer[i] = 0;
         buffer2[j] = atoi(buffer);

//     printf("pos_set : %3d %3d %3d %3d \n", buffer2[0], buffer2[1], buffer2[2], buffer2[3]);

        ps1 = buffer2[0];
        ps2 = buffer2[1];
        ps3 = buffer2[2];
        ps4 = buffer2[3];
    }
    fclose(fp);
```

```
//    exit(0);
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// Ucitavanje parametara iz FLASH-DISK file 'set'
    i=0; j=0;
    if ((fp = fopen("/var/www/set","r")) == NULL)
    {
       printf("file '/var/www/set' not found! \n");
    }
    else
    {
       while(1)
       {
         c = fgetc(fp);
         if (feof(fp)) { break; }
         buffer[i] = c; i++;
       }
       buffer[1] = 0;
       set = atoi(buffer);
//       printf("set : %d \n", set);
    }
    fclose(fp);

//    exit(0);

// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

       // Obrada - pozicioniranje antena na zadatu poziciju

    if (set==1)
    {
      set = 0;

       // - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
    //Upisi '0' u FLASH-DISK file 'set'
    if ((fp = fopen("/var/www/set","w")) == NULL)
    {
       printf("file '/var/www/set' not open! \n");
    }
    else
    {
       fputc( '0', fp);
    }
    fclose(fp);


      if      (p1 < ps1 - delta) { out1 = 1; }
      else if (p1 > ps1 + delta) { out1 = 2; }
      else { out1 = 0; }

      if      (p2 < ps2 - delta) { out2 = 1; }
      else if (p2 > ps2 + delta) { out2 = 2; }
      else { out2 = 0; }

      if      (p3 < ps3 - delta) { out3 = 1; }
      else if (p3 > ps3 + delta) { out3 = 2; }
      else { out3 = 0; }

      if      (p4 < ps4 - delta) { out4 = 1; }
      else if (p4 > ps4 + delta) { out4 = 2; }
      else { out4 = 0; }
    }
    else
    {
      if ((p1 >= ps1) && (out1==1)) { out1 = 0; }
      if ((p1 <= ps1) && (out1==2)) { out1 = 0; }
      if (out1 !=0)
      {
        time1++;
        if (time1 > time_max1) { out1 = 0; }
      }
      if ((p2 >= ps2) && (out2==1)) { out2 = 0; }
      if ((p2 <= ps2) && (out2==2)) { out2 = 0; }
      if (out2 !=0)
      {
        time2++;
        if (time2 > time_max2) { out2 = 0; }
      }

      if ((p3 >= ps3) && (out3==1)) { out3 = 0; }
      if ((p3 <= ps3) && (out3==2)) { out3 = 0; }
      if (out3 !=0)
      {
        time3++;
        if (time3 > time_max3) { out3 = 0; }
      }

      if ((p4 >= ps4) && (out4==1)) { out4 = 0; }
      if ((p4 <= ps4) && (out4==2)) { out4 = 0; }
      if (out4 !=0)
      {
        time4++;
        if (time4 > time_max4) { out4 = 0; }
      }
    }

// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// Postavnjanje izlaza
    if (out1 == 1)
    {
      bcm2835_gpio_write( 4,1);
      bcm2835_gpio_write(17,0);
    }
    else if (out1 == 2)
    {
      bcm2835_gpio_write( 4,0);
      bcm2835_gpio_write(17,1);
    }
    else
    {
      time1 = 0;
      bcm2835_gpio_write( 4,0);
      bcm2835_gpio_write(17,0);
    }

    if (out2 == 1)
    {
      bcm2835_gpio_write(18,1);
      bcm2835_gpio_write(22,0);
    }
    else if (out2 == 2)
    {
      bcm2835_gpio_write(18,0);
      bcm2835_gpio_write(22,1);
    }
    else
    {
      time2 = 0;
      bcm2835_gpio_write(18,0);
      bcm2835_gpio_write(22,0);
    }

    if (out3 == 1)
    {
      bcm2835_gpio_write(23,1);
      bcm2835_gpio_write(24,0);
    }
    else if (out3 == 2)
    {
      bcm2835_gpio_write(23,0);
      bcm2835_gpio_write(24,1);
    }
    else
    {
      time3 = 0;
      bcm2835_gpio_write(23,0);
      bcm2835_gpio_write(24,0);
    }

    if (out4 == 1)
    {
      bcm2835_gpio_write(25,1);
      bcm2835_gpio_write(27,0);
    }
    else if (out4 == 2)
    {
      bcm2835_gpio_write(25,0);
      bcm2835_gpio_write(27,1);
    }
    else
    {
      time4 = 0;
      bcm2835_gpio_write(25,0);
      bcm2835_gpio_write(27,0);
    }
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
    //Upisi podatke u RAM-DISK file 'position'
    if ((fp = fopen("/var/tmp/position","w")) == NULL)
    {
       printf("file '/var/tmp/position' not open! \n");
    }
    else
    {
       fprintf(fp,"%3d,%3d,%3d,%3d\n",p1,p2,p3,p4);
    }
    fclose(fp);
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
    printf("%d = Out1: %3d %3d %d %3d - Out2: %3d %3d %d %3d - Out3: %3d %3d %d %3d - Out4: %3d %3d %d %3d
\n",set,ps1,p1,out1,time1,ps2,p2,out2,time2,ps3,p3,out3,time3,ps4,p4,out4,time4);

    bcm2835_delay(250);  // delay 0.25sec

  }
    bcm2835_spi_end();
    bcm2835_close();
    return 0;
}
```

REFERENCES:

http://www.digistore.rs/
http://www.raspberrypi.org/
http://wiringpi.com/