Data

|  |  |  |
| --- | --- | --- |
| Data | Type | Values |
| Select position | unsigned char | 0 - position A 1 - position B 2 - position C 3 - position D etc... |

Settings

Settings of "select" allow to specify the different values:

* **Variable name** - name of the select and the variable in the source code for the microcontroller, allow to set a name for the variable of C rules.
* **Count of positions** - sets the number of possible positions of the "select". Can take a value from 2 to 10.
* **Orientation** - You can set the orientation of the "select" on the interface field. Orientation can be "vertical" or "horizontal".

A common example showing how to check the current position of the four position switch:

if (RemoteXY.select\_1==0) {

/\* current position A \*/

}

else if (RemoteXY.select\_1==1) {

/\* current position B \*/

}

else if (RemoteXY.select\_1==2) {

/\* current position C \*/

}

else if (RemoteXY.select\_1==3) {

/\* current position D \*/

}

Code with two on of switches and a position switch

/\*

-- New project --

This source code of graphical user interface

has been generated automatically by RemoteXY editor.

To compile this code using RemoteXY library 3.1.11 or later version

download by link http://remotexy.com/en/library/

To connect using RemoteXY mobile app by link http://remotexy.com/en/download/

- for ANDROID 4.11.4 or later version;

- for iOS 1.9.1 or later version;

This source code is free software; you can redistribute it and/or

modify it under the terms of the GNU Lesser General Public

License as published by the Free Software Foundation; either

version 2.1 of the License, or (at your option) any later version.

\*/

//////////////////////////////////////////////

// RemoteXY include library //

//////////////////////////////////////////////

// RemoteXY select connection mode and include library

#define REMOTEXY\_MODE\_\_SOFTSERIAL

#include <SoftwareSerial.h>

#include <RemoteXY.h>

// RemoteXY connection settings

#define REMOTEXY\_SERIAL\_RX 2

#define REMOTEXY\_SERIAL\_TX 3

#define REMOTEXY\_SERIAL\_SPEED 9600

// RemoteXY configurate

#pragma pack(push, 1)

uint8\_t RemoteXY\_CONF[] = // 54 bytes

{ 255,3,0,0,0,47,0,16,31,1,3,10,39,14,6,58,2,26,10,48,

11,24,15,15,4,26,31,85,80,0,31,79,70,70,0,10,48,11,44,15,

15,4,26,31,68,79,87,78,0,31,79,70,70,0 };

// this structure defines all the variables and events of your control interface

struct {

// input variables

uint8\_t select\_1; // =0 if select position A, =1 if position B, =2 if position C, ...

uint8\_t pushSwitch\_1; // =1 if state is ON, else =0

uint8\_t pushSwitch\_2; // =1 if state is ON, else =0

// other variable

uint8\_t connect\_flag; // =1 if wire connected, else =0

} RemoteXY;

#pragma pack(pop)

/////////////////////////////////////////////

// END RemoteXY include //

/////////////////////////////////////////////

#define PIN\_PUSHSWITCH\_1 13

#define PIN\_PUSHSWITCH\_2 13

void setup()

{

RemoteXY\_Init ();

pinMode (PIN\_PUSHSWITCH\_1, OUTPUT);

pinMode (PIN\_PUSHSWITCH\_2, OUTPUT);

// TODO you setup code

}

void loop()

{

RemoteXY\_Handler ();

digitalWrite(PIN\_PUSHSWITCH\_1, (RemoteXY.pushSwitch\_1==0)?LOW:HIGH);

digitalWrite(PIN\_PUSHSWITCH\_2, (RemoteXY.pushSwitch\_2==0)?LOW:HIGH);

// TODO you loop code

// use the RemoteXY structure for data transfer

// do not call delay(), use instead RemoteXY\_delay()

}