

Circuit Cellar Atmel AVR Contest 2004 Project

A Reliable Low-cost
RADIO PACKET CONTROLLER

Based on Bascom-AVR and the Atmel AT90S2313 Microcontroller

(Abstract)

Project No: A3522
Name: Satz Pillai
Country: Singapore

(A project for the intermediate-level experimenter)

A Reliable Low-cost Radio Packet Controller - Abstract

1. Summary

This project describes the software design of a Radio Packet Controller (RPC) built around an Atmel AVR 8-bit flash micro-controller programmed entirely in Bascom-AVR Basic. When interfaced to a host controller, it allows reliable half-duplex serial RF data communication among a maximum of fifteen devices in a simple, low-speed radio network. The RPC handles all low-level encoding & decoding and error checking and its modular design allows RPC characteristics to be easily adapted to suit other applications like RF repeaters, data loggers, transponders, remote control, and so on. The assembled RPC module has a deliberate “homebrew” appearance, and is meant to demonstrate that the project is well within the reach of the intermediate level experimenter.

2. Motivation

The idea for this project grew out of another project to design a group of ‘remote-brained’ robots operating under central control, the concept being that the robots’ limited processing capability could be enhanced by providing additional centralized processing power in a PC base-station. To enable this, a reliable two-way RF link was required, where data packets could be exchanged between the robots and the base-station.

Sending and receiving packets over the RF medium poses special problems due to the effects of noise, interference, transmitter & receiver characteristics and variations in signal strength, etc. To ensure reliable communication, a suitable RF communication protocol must be defined, transmitters and receivers have to be synchronized and packets have to be encoded and decoded with error detection & correction schemes built in. All this requires processing power, which is in short supply in a robot controller. The job is best managed by having a dedicated radio packet controller to handle all packet transmission and reception tasks, thereby freeing the robot (or base-station) controller for other tasks. Since every robot and the base-station would need its own RPC, a cost-effective solution is essential.

Surprisingly, there are not many commercial products that fill this need, possibly due to low demand. Products that do exist suffer from inherent disadvantages: (i) high cost, (ii) bulky and power-hungry (RF modems), (iii) interfacing complexity (bluetooth devices), (iv) limited capability (encoder/decoder ICs), (v) limited flexibility (transceiver controller ICs). Given these disadvantages, it made sense to build one using suitable low cost components.

Although originally designed for the robot project, the RPC clearly had application potential in other areas; and so the design was modified to make it into a building block for general-purpose use, the design and construction of which is highlighted in this document.

3. RPC Features

- Allows half-duplex communication among a maximum of fifteen addressable nodes.
- 4 selectable operational modes: (i) Transmit & Receive, (ii) Receive, (iii) Transmit, (iv) Test
- Maximum data packet size of 52 bytes, with a data payload of 50 bytes
- Packets can be addressed to specific nodes or broadcast to all
- RPC acknowledges all non-broadcast packets addressed to it
- Automatic retransmission of unacknowledged packets, up to a user configurable maximum
- Manchester encoding/decoding of data to ensure communication reliability
- Manchester encoding/decoding and basic CRC8 error checking are transparent to user
- Operates on a single 5V supply at 15mA average, with power save mode to conserve power
- Can be interfaced directly to a host microcontroller working on 5V CMOS logic
- Can be used with transceivers, or with separate transmitter/receiver modules; in-building and open air range are dependant on the RF modules used
- RPC code can be easily ported to other controllers in the AVR family

A Reliable Low-cost Radio Packet Controller - Abstract

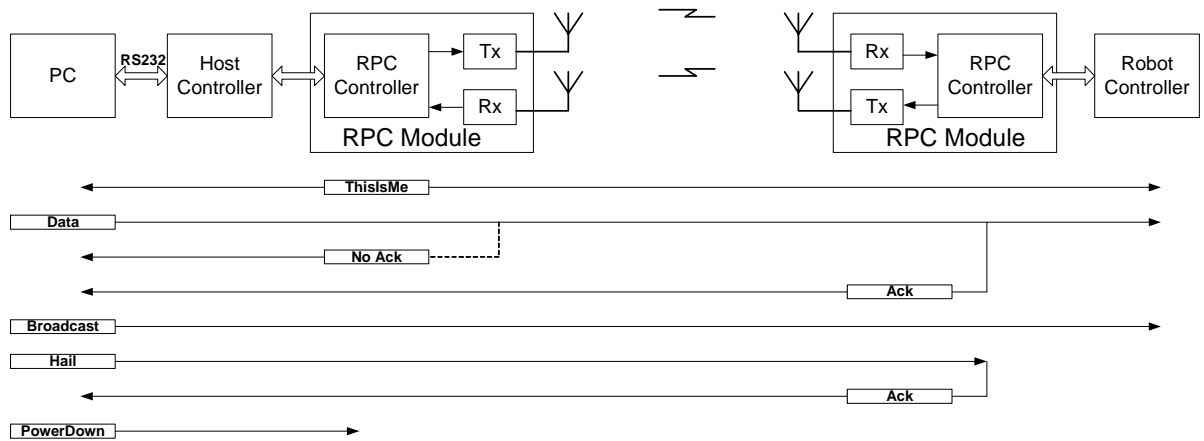


Figure 1 RPC Block Diagram

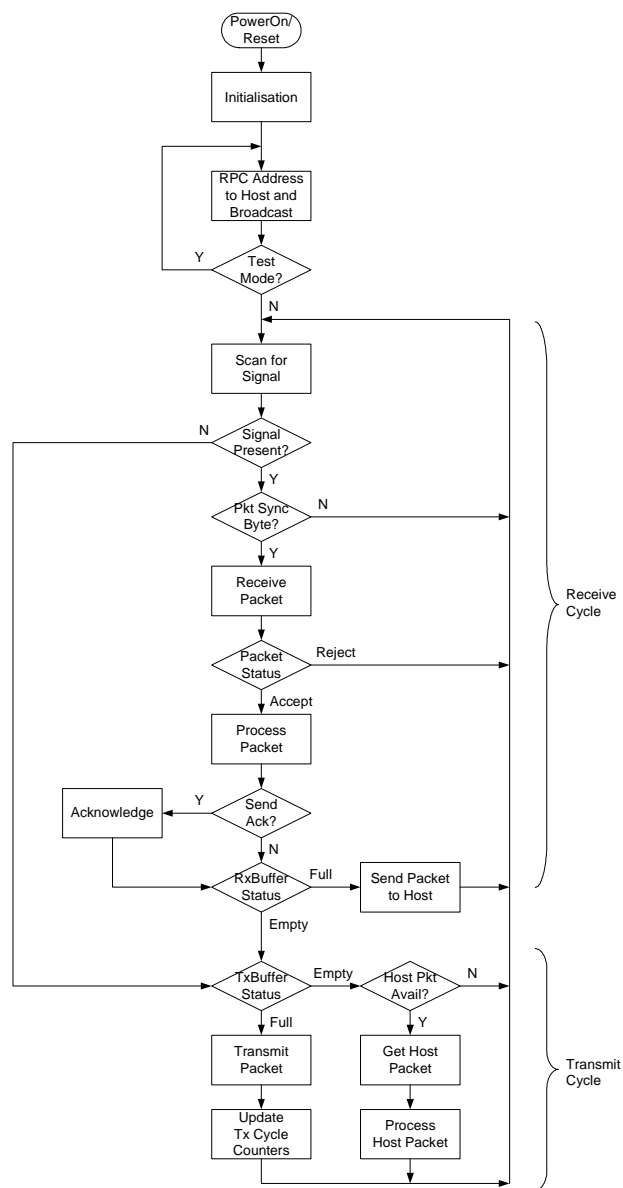


Figure 2 RPC Control Logic Flowchart

A Reliable Low-cost Radio Packet Controller - Abstract

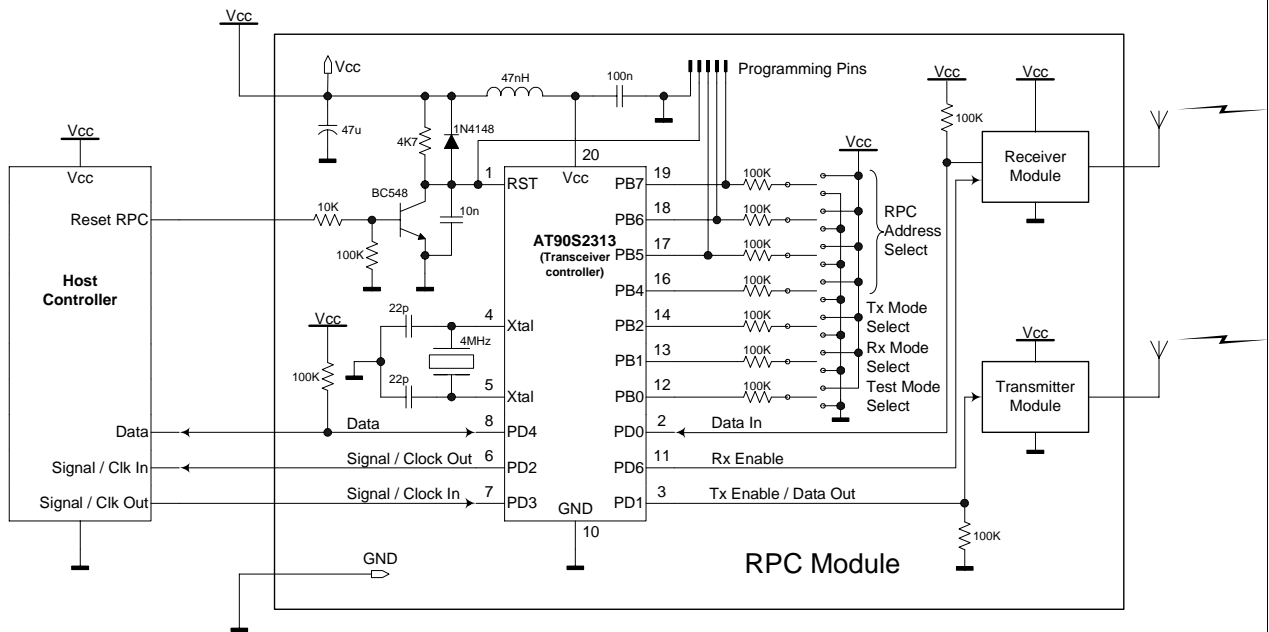


Figure 3 RPC Schematic Diagram

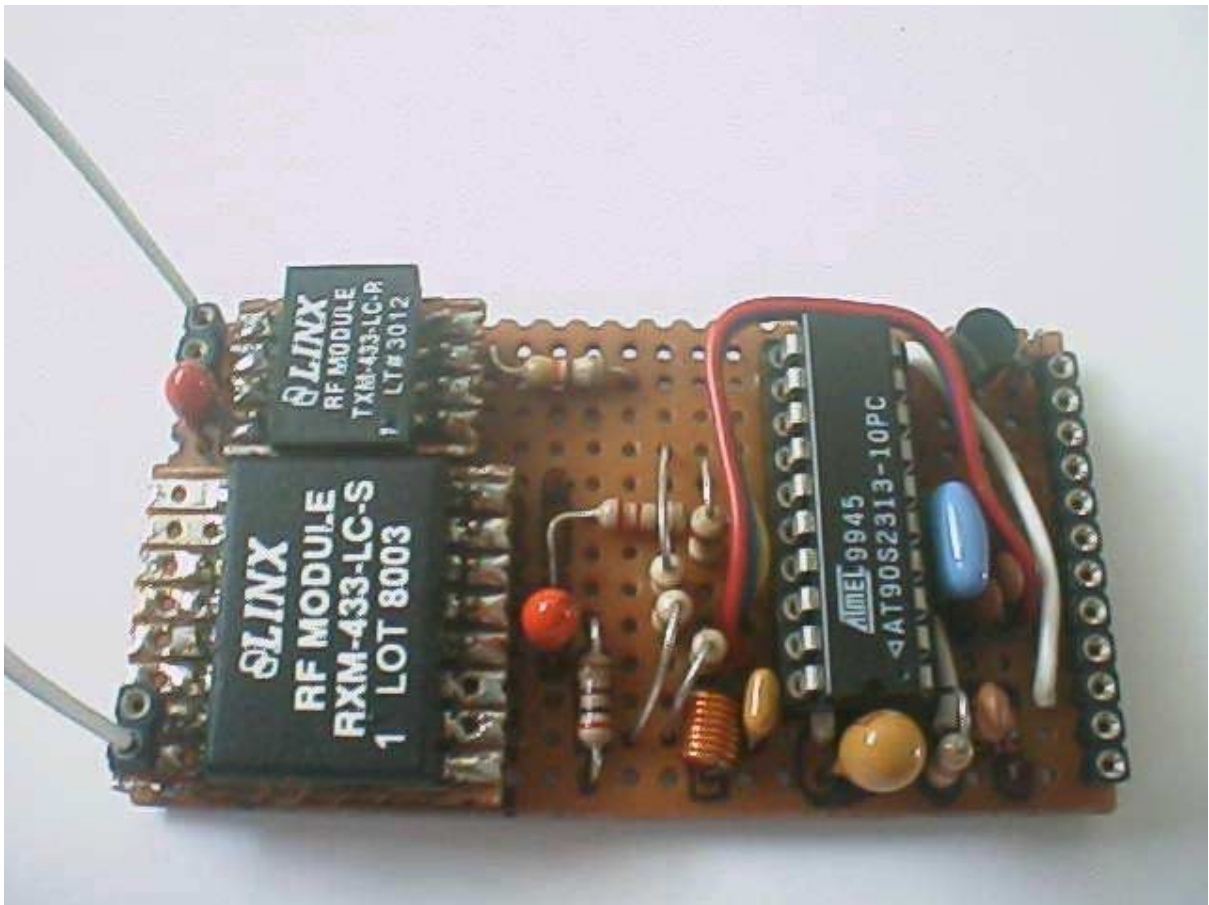


Figure 4 Fully Assembled RPC Module

A Reliable Low-cost Radio Packet Controller - Abstract

Sample Program Listing

```
'=====
' Program Name : RPC.bas
' Description  : A Reliable Low Cost Radio Packet Controller (RPC)
' Entry ref #  : A3522
' Author       : Satz Pillai
' Compiler     : Bascom-AVR ver 1.11.7.4 (Demo version available at http://www.mcselec.com/download\_avr.htm)
'=====

Main:                                'RPC main control module
  Do                                  'While RPC is in normal mode
    If RxMode = Active Then          'If Rx mode is active,
      Gosub ReceiveProcessScheduler '   Receive packet & send to host cycle
    End If
    If TxMode = Active Then          'If Tx mode is active
      Gosub TransmitProcessScheduler '   Get packet from host & transmit cycle
    End If
  Loop

ReceiveProcessScheduler:              'Receive packet and send to host
  Do                                  'Start cycle
    Gosub ScanForSignal               'Scan for carrier signal
    If ReceivedByte = StartOfPacketSync Then 'If start-of-packet sync byte is detected,
      Timeout = TimeoutNormal
      Gosub ReceivePacket             '   then receive packet
      If RxPacketStatus = Accept Then '   If packet has been accepted,
        Gosub ProcessReceivedPacket '     process packet
      End If
    End If
  Loop Until SignalDetect = No        'Repeat cycle until no signal is detected
  Return

TransmitProcessScheduler:             'Gets host packet if Tx buffer is empty, Else transmits
  If TxBufferStatus = Empty Then      'If Tx buffer is empty &
    If SignalIn = RequestPacketTransfer Then 'If Host has initiated communication,
      Gosub GetPacketForTx            '   Get packet from host
      Gosub ProcessPacketForTx        '   Process packet for transmission
    End If
  Else                                  'If Tx buffer is full then,
    Gosub TransmitPacket              '   Transmit packet
    Gosub PacketRetransmitControl     '   Packet retransmit control
  End If
  Return
```