Implementation notes and Docs for an Ethernet solution usable for embedded systems using the RTL8019(AS) chip

Notes and Docs collected by Werner Cornelius (Cornelius@ethernet.isdn-development.de)

Some time ago when I have been searching for an ethernet controller suitable for embedded applications using standard microcontrollers like the 8051s or a NEC V25. I found the RTL8019AS chip manufactured by Realtek Taiwan (www.realtek.com.tw).
This chip offers some advantages to other available solutions:

1. A RAM buffer with 16KBytes size is included on the chip and needs no external wiring. This RAM is completely controlled by the chip and offers buffering for multiple packet data . The data is aligned to 256 byte boundaries, so every received packet requires at least 256 bytes. If one send buffer for an ethernet packet with 1576 bytes is assumed (occuping 1792 bytes due to alignment) and a maximum packet size of 1576 bytes is expected for receive at least 8 received packets with full size may be buffered. If the received packets are smaller of course more packets may be buffered.
2. The chip is software compatible to the popular NE1000/NE2000 hardware offering 16 and 8 Bit access modes. So drivers may be easily adapted or written.
3. There are only a few components required for using the chip, as the complete address decode is included (10 or 16 address bits possible for decode).
4. The chip is suitable for 10 Base T (twisted pair), BNC and AUI connections. A media auto detect is build in.

The only disadvantage may be the QFP100 package used, which is a bit complicated to solder but with its 0.65mm spacing this might be achived if you have some experience.

Before creating a own board a test with an ready manufactured PC card using this chip might be a quick check whether this solution is suitable for the desired purposes. Wires connected via an connector or even soldered directly to the card edge connector may be used for first tests.

I/O-Address decoding

The RTL 8019AS has an included address decoder for minimal external hardware requirements. Additionally the address may be changed inside the running system via "Plug and Play" mechanism, which is normally not needed in microcontroller applications.
The I/O address decoder setup is done during 2ms after a hardware chip reset. The setup data is fetched from the external EEPROM or the jumpers if set to jumper mode.
A common pitfall when using the device with the internal address decoder is that there are 2 possible modes of decoding selectable:

1. The chip only decodes addresses A0 to A9. This mode is set when bit 0 of byte offset 3

inside the EEPROM has a zero value.
2. The chip decodes the I/O-address line range using A0 to A15. In this case the bit must be set to 1

The mode is determined at reset when reading the EEPROM and MAY NOT be changed by the user afterwards. If an EEPROM with this bit reset to 0 is used and all 16 address lines required or used to decode some problems may occur.
The EEPROM may be reprogrammed inside the system if needed.
The following binary is taken from a card using full 16 address line decode.
The Plug and Play data inside the EEPROM may be used to save additional parms like IP-addresses and other user config data, when the Plug and Play function normally not needed isn't used at all.

Sample EEPROM config data

93c46.bin

How to connect the card/chip to a 8051 system:

I assume you will connect the card in the external RAM-space

1. Connect the data lines and addresses A0-A4 directly to the RAMs addresses
2. Set the card to jumper mode, with the following setup:
   An irq of your choice
   Boot ROM disabled !!!
   Select a base addresses selecting from the following criteria:

   If you would like the RTL8019 to completely decode your address space
   think about the area you would like to use.
   Then select a jumpered base address (0x200-0x3e0) which has the needed number of 1 bits as the base (1 to 5 bits possible) and connect the rest of the address lines apropriately.

   Example1: You would like to use the standard base address 0x200
   Then connect the remaining address lines A5-A15 to the same ones on the RAM.

   Example2: As 200 to 3e0 hex is normally used for RAMs you might like to occupy the chip the high 2 KBytes of the RAM space.
   Set the jumpers to base address 0x3E0
   Connect A5-A9 to A15-A11 of the RAM and the rest of the cards address lines A10-A15 to A5-A10 of the ram.
   The card then will occupy addresses 0xF800-0xFFFF each 32 bytes duplicated of course.

In this both examples the /AEN line of the connector needs to be grounded !

If you want to use an already existing Address decoder with an active low output inside your system, setup a desired address with the jumpers and connect A15-A5 to the appropriate levels for the card to be decoded.
Connect the /AEN line to your decoder.

For all examples the cards addresses A19-A16 and the /MEMCS16 lines should be grounded !
/IORD and /IOWR of the card are connected to the 8051s /RD and /WR.
/MEMRD and /MEMWR must be pulled high because else the card may use the ready line not supported by any 8051 !!!
Now only the positive Reset input for the card may be connected to a reset on the 8051 system or may be better connected to a single pulled up port line, allowing hardware reset of the card in the running system.
(After a hardware reset wait at least 2ms before accessing the card)
The selected interrupt line from the jumpers may be connected via an inverter to an 8051s interrupt line. This line of the card is active positive, so the inverter is required. Edge or level irq may be used, depending on the implemented software.

Now all connections are complete !
You may test the circuit.

How to connect the card/chip to a V25 system:

Mainly the same thinsg regarding address decoding as above apply but there are a few differences:

1. The /AEN line of the controller needs to be connected to /IOSTB of the V25
2. The /IOWR line of the controller is connected to /WR of the V25
3. The /IORD line needs to be an inverted signal from the V25 /WR
4. The reset signal may be derived from the inverted V25s /RESET or a positive RESET line available in the system. Also the solution to connect it to a port line (pullup required) is suitable.
5. I have used the V25+ with 10 MHz and 2 wait states for the io access. In this case the RTL chip never generated a wait on the ready line, so connecting this line is not needed. Boot rom must be disabled and /MEMWR and /MEMRD pulled up of course.

Dokumentation and data sheets

For the first I may offer the following documents I got from Realtek:
Please send me additional docs you got or wrote.

Data sheets:

RTL 8019 data sheet (version without internal RAM)
rtl8019.pdf

RTL 8019AS addendum to RTL 8019 data sheet (with internal RAM)
8019asds.pdf

Schematics (For Orcad schematic (DOS version))

2 in 1 (An ISA card offering 10 Base T (twisted pair) and BNC
2in1.zip

3 in 1 (Same as above but additionally with auto sense AUI connection)
3in1.zip

Software

Packet driver coded in 8086 asm adapted from NE1000/2000 driver for use with RTL8019(AS) chip by Realtek.
Please tell me if additional software may be put here.
pkt8019.zip

I hope this documentation will help to implement new designs.
In the meantime I spoke to Realtek and I decided to supply interested developers in germany (other locations on request) with chips in small sample quantities, as it would not be a good idea to order and sent 5, 10 or 20 pieces directly from Taiwan to Europe. If you need samples please contact me and I will give you the details.

Werner Cornelius
E-Mail to: Cornelius@ethernet.isdn-development.de