Re: Knowledge in DUP−line protocol?

Source: http://sci.tech−archive.net/Archive/sci.electronics.design/2007−11/msg03059.html

- *From*: JosephKK <joseph_barrett@xxxxxxxxxxxxx>
- *Date*: Mon, 19 Nov 2007 20:20:53 −0800

Run.PDP nnc@xxxxxxxxxxxxxxx posted to sci.electronics.design:

> On 12 Nov, 18:45, JosephKK <joseph_barr...@xxxxxxxxxxxxx> wrote:
>
> > Run.PDP n...@xxxxxxxxxxxxxxx posted to sci.electronics.design:
> >
> > > On 11 Nov, 02:20, Jamie
> > > <jamie_ka1lpa_not_valid_after_ka1l...@xxxxxxxxxxx>
> > > wrote:
> > >
> > > > Run.PDP wrote:
> > > >
> > > > > Anyone that has deeper
> > > > > knowledge in
> > > > > theDUPlineserial protocol?
> > > >
> > > > As to my knowledge,
> > > > High−level is 8,2 V.
> > > > Low−level is 2,2V or
> > > > less Channel−Generator =
> > > > Master−Generator by
> > > > "inactivity" =
> > > > all bits zero sends out a
> > > > continuous sync train,
> > > > StartOfTransmission pulse
> > > > =8 ms of Low, followed by
> > > > N (32, 64,
> > > > 128) pulses at time interval
> > > > 1 ms, each pulse Low for
> > > > 0,3 ms,
> > > > High for 0,7 ms, to describe
> > > > the 0−signal.

When 1 is transmitted, this is done by reversing the actual
pulse, Low for 0,7 s, High for 0,3 s.

When a device on the bus wants a bit to become 1, it will
short the bus (typical voltage is 0,7V during the short).
This will be noted by the MG / CG, that answers with a regular
1 sequence.

BUT:
How does the MG recognince that someone else wants this bit to become a 1?
Is it the lowering in voltage (from 2,2V to 0,7V), or is it the signal remaining low (like < 4V) during the time of 0,3 −
0,5 ms, ie when the master tries to make a zero by rizing from
Low to High?

So, how long must an external device keep low to set the 1,
and when must
it start?

Best regards,
Göran I Åhling

I assume you're talking about multiplexing devices on the same
wire?
if so, all devices should be open−colllector outputs. The
master or
some one in the buss must provide the pull up..
Normally, all devices monitor the line for traffic to other
devices
along with the master. Since all devices are open−collector on
TX, the RX is tied to the same line and can monitor it self and
others.

No device should attempt to do any TX while a frame time of
sequences
are in process.

Is this what you're looking for?

––
"I'm never wrong, once i thought i was, but was mistaken"
Real Programmers Do things like this.http://webpages.charter.net/jamie_5

Well,.... To be polite.... This principal statement of multiplexed busses is of cause correct, and valid also for theDUP−linebus policy.

But, TheDUP−lineprotocol is allready engineered further! My
question might seem cryptic, but to a reader allready familiar with
DUP−linelow level signaling, it is all obvious! (and my

question
can not be correctely answered by someone who does not have this
low−level knowledge!). TheDUP−lineprocol is, as far as I know, a
proprietary protocol from one wendor used to distribute a number
of bits on a syncronous bus. The number of bits can be selected
for each
installation, alternatives are 32, 64 or max. 128. The bits are always transmitted with duration time of 1 ms for each bit. After
transmission of each block of bits, headed by a 8 ms starter, the
transmission is repeated.

This render in a repetition rate, depending on the number of channels selected, of 40, 72 or 136 ms. The first implementations
made were designed using standard C−MOS MSI circuits. In order to
keep cost reasonable while capabilities should be substantial and
robust, the scheme of a local clock at each attached device was
never implemented. Instead there is one "Channel Generator" (in
later designs this function is included within the "Master Generator" besides several other functions) that continuously clocks the entire bus.

The default signal sent by the CG. constitutes of a continuous
train of blocks, each one having one start/sync pulse and N blocks of 0
(zero). A 0 is sent as bringing the bus to low voltage for 0,3 ms, followed by letting it high for 0,7 ms.
If, for some reason, one channel is to be transmitted as a 1 (one),
this is signaled as 0,7 s of low, followed by 0,3 ms of high. In
this way there always comes a pulse for each ms, the falling flank of it (from high to low) marks the start of each new bit, this flank can be used to count position within each block.

As you said, the bus is open−collector. Any device connected to
this bus can pull any bit low, there by setting it to a 1, or active state. When this is done, the CG recognize this state, and
repeats it to make sure all devices on the bus get the information (full communication between any two nodes on the net
is thus not required, the only required communication path is between any node and the CG.)

I have measured and can see the voltages from CG being 8,2 V for
high, 2,2 V for low, and the bus goes like 0,7V when an external
node pulls low.

My explicit question is that if someone knows what mechanism the
CG uses to recognize that any other device want to set a bit to
1: Either to recognize the lowered voltage 0,7 V as opposed to
2,2V, or just the fact that the bus remainslow (lets say below 4
V?) at the time when
the CG. tries to rise (eg at the time 0,3 ms after the start of a each bit). Also how long time the external device needs to assert the low in order for the CG to recognize this state. As far as I know, a handshake/acknowledge is possible within the bit
(the sender pulling low can do this "early" within the bit, and get a confirmation that the CG is repeating this at a time slightly before the 0,7 ms point.

I presume there is no need to explain that lots of different formats of information can be implemented using this basic bit−carrying network as base, for example nodes for "transmitting" and "receiving" analogue information (the person
installing and configuring can select for each analogue channel
if it may use 12 of the bits and have a bandwidth equal to 1/2 of
the repetition rate or if it only may use one bit, but might

have
a bandwidth that is only 1/12 of the one above... (1/2 of the sampling rate, as stated by the sampling theorem).

This protocol has been around for at least 20 years by now, it is
widely used in different industrial applications here in Europe.
It was developed by the danish company "Electromatic", that was
later bought by "Carlo Gavazzi", who has continued developing it.
Nowadays, it is also sold as "smart−home" network. In Sweden and
Norway (it might also be in other countries that I don't know about), a "ELKOMATIC bus network is also sold by the large
manufacturer of
domestic appliances (CB:s, light switches, outlets, ...) ELKO.
This network is nothing but a OEM−edDUP−line.

Practical applications of this network is in the range of up to 10 km of network length (about 6 miles) using standard installations wire (not requiring shielding or twisting). I have no knowledge on the leagal state of this protocol, ie if it would
be criminal to start manufacture "3:d party devices" using the same protocol without bying the communication ASIC from Carlo
Gavazzi. In reality, using for example a small PIC−processor, it
would be a peace of cake to design a simple node.

As you might have concluded, the bus is sensitive to short−circuits, which will render in all receivers recognizing every bit as 1 or active. The bus also has a weakness in that you
can not know if some devices "at the far end of a branch" might
have been dis−connected unless a transmitter is connected at the
very end of each branch. This transmitter should be allocated a
unique channel where it continuously sends a 1, or even better

sends a slow oscillator signal so that some central device can monitor connection and working transmitter.

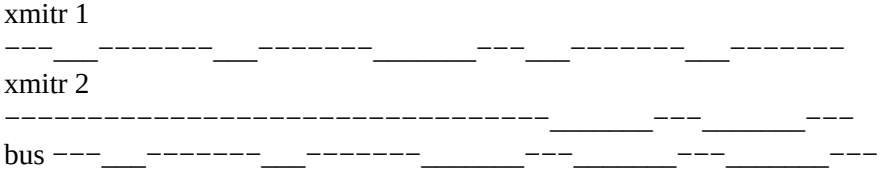
With these low voltages, one might think the bus should be sensitive to cross−talk interference, but practical implementations show quite the opposite. The manufacturer has
improved their devices in regards of withstanding over−voltage
(Thunder...) over time. The entire structure has a reputation of
being slightly simple, being very easy to install (can even be designed by the installer, an engineering consultant is not needed!), and being utterly robust in operations.

The reason for my questions is that I'm considering playing a little with this protocol in controlling some lights, some outlets, some temperatures, etc in my house, but I'd like to implement a few functions myself, and to combine this with buying
some power−switching devices ready−made (with formal approval and
all). My first round of implementation would be using the parallel port of an old lap−top running W.95 and Borland Pascal/Delphi.

Best regards / 73

Göran I Åhling / SM6NNC

I do not understand your difficulty. Any device on the bus can change
any 0 to a 1. All devices on the bus compare what is on the bus to when they are trying to transmit, and will thus know.



Thus xmtr 1 can see that it is being interfered with. Now where

> > are these request bits placed in the transmitted block?
>
> This is true for a generic "Wired or" bus structure.
>
> DUP−line is much more than just two pull−upped wires and some transistors.
>
> Amongst other things, there is a "Bus−master" (so called master−generator or channel−generator) that is continuously controlling the bus by sending frames, each consisting of a start−block and N bits of information. By default, these bits are 0, but any device can influence on the channel−generator for any bit, so that the transmission of a 0 is changed "on the fly" to a transmission of a 1.
>
> My questions are addressed to anybody that has the explicit knowledge in this bus technology, its timing spec etc.
>
> Best regards /Göran

Then why didn't you read what i posted. I posted how to do part of the solution that you said you were missing. I also directed you to where to look for for the rest of the solution.

.