DALLAS SEMICONDUCTOR

Application Note 106
Complex MicroLANs

I. GENERAL CONSIDERATIONS

A. Topology

The MicroLAN™ is a feeder network using a single data line plus ground reference for digital communication. It provides digital information at very low cost to computers and their networks. A MicroLAN (Figure 1) can be likened to a tree consisting of a trunk and many branches. Its trunk connects via a 1–Wire™ driver to an RS232 COM port of a Personal Computer which can act as a MicroLAN server to a larger network. This computer is also referred to as Bus Master. It runs TMEX 1–Wire communication software under MS–DOS or WINDOWS.

At the most distant end of the trunk, an iButton called Trunk Marker is attached. Along the trunk there are addressable switches that branch to extensions called limbs. The limbs also have their markers at the ends of the cables. Branches extending from a limb are called twigs, branches extending from a twig are called stems. In any case, each branch point has its own marker at the most distant end of the path. The function of markers is to verify that communication can reliably pass all the way to the path extremities. Markers are distinguished from other devices by their data contents. In the general case, mobile iButtons (the sub–network), sensors and I/O switches are connected like leaves to limbs, twigs and stems, but not to the trunk. If a network has no branches, however, the “leaves” are directly connected to the trunk.

To communicate with a device on a twig, first the addressable switch connecting the limb to the trunk and then the switch connecting the twig to the limb must be activated. Activating branches with software controlled switches allows the building of huge feeder networks, yet avoids having all devices loading the cable at the same time. In addition, the addressable switch can provide information about physical location. This is not possible if all components are residing on the trunk.

Data communicated via the MicroLAN is packetized. Either the MicroLAN device adds a CRC (cyclic redundancy check) to the data or the data itself is packetized before it is written to a device. If for any reason a packet gets corrupted during data transfer, the Bus Master can detect the error by checking the CRC and repeat the data transfer. (See the “Book of DS19xx iButton Standards” for details.)

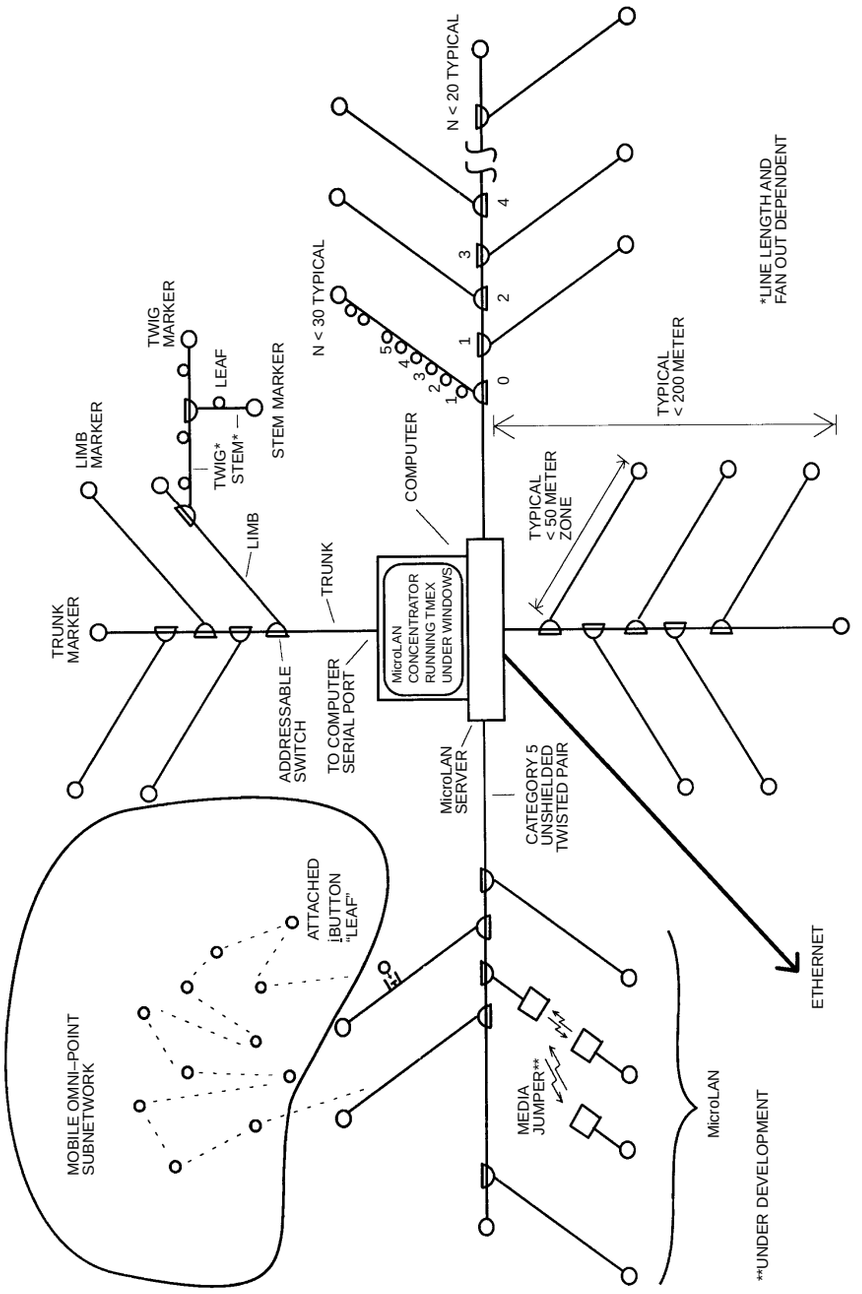
Communication on MicroLAN branches is not event driven; the Master typically polls the nodes and checks if an activity is required. A Personal Computer equipped with several COM ports can support several independent MicroLANs (trees), thereby increasing the performance or total size of the feeder network.

B. Building Branches

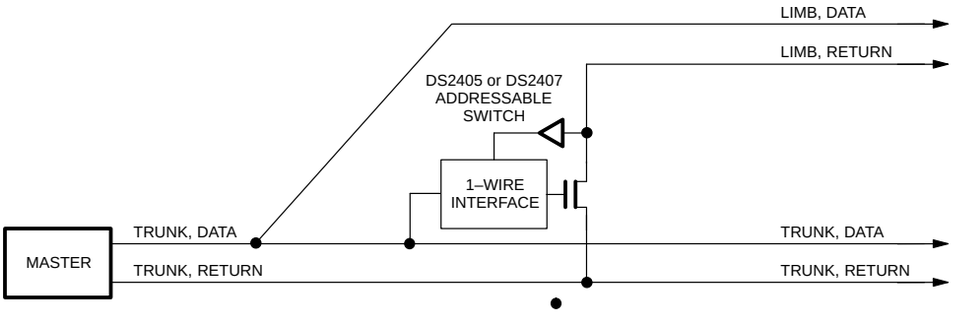
To realize huge feeder networks without simultaneously increasing the electrical load, MicroLAN uses a tree–like topology consisting of several levels of branches that finally connect to the trunk. The core component to activate and deactivate branches (i. e., making connection under software control) is the addressable switch. This device is basically a transistor that can be remote–controlled (switched on/off or sensed) via its MicroLAN interface. How the addressable switch connects a limb to the trunk (or creates a new branch level somewhere in a MicroLAN) is detailed in Figure 2. The Data Line of the MicroLAN is permanently routed to all devices in the system. The Return Line, however, is conducting only in those branches that are needed to access the leaves the master intends to communicate with.

Note that the MicroLAN Return Line is not identical to system ground. To avoid ground loops, a circuit employing optoisolators is required to communicate with circuits that must be grounded. Figure 3 gives an example of such an optoisolated interface.

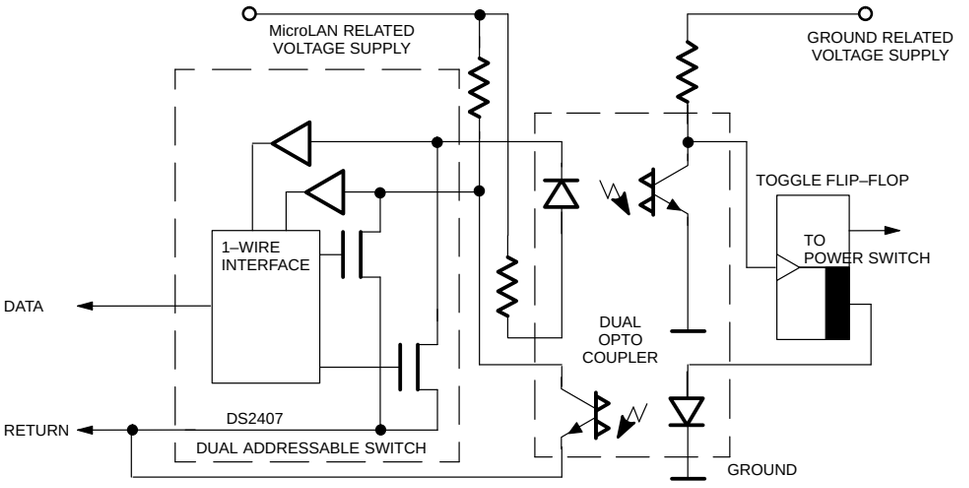
UNLIMITED NETWORKING Figure 1



BUILDING BRANCHES Figure 2



OPTO–ISOLATED INTERFACE Figure 3



C. Software Considerations

After powering–up a MicroLAN, the bus master will see only those 1–Wire devices that reside on the trunk. In the general case these are addressable switches and the trunk marker. Before any communication beyond the trunk can take place the bus master needs to learn the topology of the network. The first step therefore is to perform an analysis of only the Addressable Switches on the MicroLAN. Beginning with the Trunk, the bus master systematically opens the switches to every limb one by one and records the registration numbers of the switches that it finds, then opens those switches one by one, etc., until it has built a complete model of the network topology in memory. In the second step the master needs to identify all other devices on the MicroLAN that are not addressable switches. Beginning with the Trunk and using the internal tree developed in the first step, the bus master systematically opens the switches to every limb one by one and records the registration numbers of the devices that it finds which are not Addressable Switches until it has added all additional devices to the memory model of the network topology. As the master knows the topology and population of the MicroLAN it now is able to build the path to any device in the network and to communicate with it. After the master determines which device the user wants to access it opens the switches necessary to reach the device, and addresses it with a Match ROM command. At this point, all devices have been deselected except the target device, which is prepared to accept a memory function command. This

scheme allows data to be read from or written to any iButton in the network, regardless of its position.

Software like this has already been developed by Dallas Semiconductor and will be available as TMEX MicroLAN Manager, part number DS0630N. This software is designed for the Microsoft Windows operating environment as a Dynamic Link Library which the software developer can link with his Windows programs. The MicroLAN Manager supports MicroLANs with rigid as well as permanently changing topology. An example of changing topology is a network with docking stations where sub–networks may connect at any time. Such a sub–network could be a cart with containers that are wired as a MicroLAN themselves.

II. IMPACT OF COMPONENTS

A. Effects From the Cable

With long distances, any cable shows transmission line effects unless the signal slew rate is slow compared to the signal propagation time on the cable. Undesirable signal excursions (reflections) which can disrupt communications show up if the cable is not properly terminated and the transition time of the transmitted signals is too short for the electrical length of the cable. In the case of the MicroLAN it is not possible to terminate the cable, so controlling the signal slew rate is important.

Each cable type has its own characteristic capacitance, inductance and resistance. These parameters are determined by the cable geometry, the type of conductor (solid, stranded, gauge size, etc.) and the type dielectric between the conductors. These define the characteristic impedance of the cable, the signal damping and the signal propagation velocity which for a practical cable is approximately 2/3 the speed of light. Typically, the specific capacitance which can range from 30 pF/m to 100 pF/m, needs to be taken into account since it contributes significantly to the total capacitive load of a network. Also the differential inductance of the cable becomes increasingly important as cable length increases.

The ohmic losses inside the cable reduce the noise margin of the digital signals. When building large networks, the branches and addressable switches connecting them add to the ohmic losses of the trunk. Usually the cable capacitance together with the parasitic power supply of the 1–Wire devices are the major factors that limit the size of the network. This requires a precharge time of a few milliseconds before communication can start on the MicroLAN, especially if a passive (resistive) pull–up is used for providing energy and communication voltage. A precharge time of one millisecond should also be allowed if one activates a new branch of the network. The parasitic power supply of the new arriving devices will cause a droop in the 1–Wire idle voltage until the energy reservoirs of the new devices are filled sufficiently. After that, the recovery time after each time slot will be long enough to replenish energy used during the previous time slot.

B. Open Drain Considerations

The MicroLAN is an open drain (wired–AND) environment with typically a passive resistor pull–up to the operating voltage of nominally 5V. Communication is done in time slots of 60 µs minimum per bit plus a short recovery time between time slots. Due to the low impedance active pull–down to generate a logic 0, the fall time of the communication signals is very short. The rise time is determined by the product of pull–up resistor and the total capacitive load of the active branches of the network, counting the cable itself and the input capacitance of the devices. The maximum high voltage on the MicroLAN is determined by the value of the pull–up resistor and the total idle current of all devices on the activated segments of the network. The more devices that are listening, the higher the voltage drop across the pull–up resistor will be, limiting the maximum voltage the network can reach. The higher the voltage drop, the longer it will take for the network to reach the logic 1 level of 2.2V and the minimum operating voltage of 2.8V. This has some consequences for the 1–Wire communication.

The critical part of 1–Wire communication is the read data time slot, especially if a 1 is being transmitted. This becomes most critical in the ROM search process, which is required to identify the devices on the bus. In the general case, there may be many switches scattered on the trunk as well as leaves on the selected limb and twig and stem. Each of these devices sees the trigger provided by the falling edge of the time slot issued by the master at a slightly different time point (variations up to 2 µs due to transmission line delay). Theoretically, any device may have a cycle time (t_{RDV}) as short as 15 µs. Since the position of the devices on the limbs and trunk is arbitrary and the cable can never be terminated with its characteristic impedance echoes in the form of spikes and undershoots may occur unless the master's

pull down transistor has its LOW–going slew rate limited.

The rise time of the communication signals can be improved by reducing the value of the pull–up resistor, by using lower capacitance cable, or reducing the load of devices in the active segments of the network, whichever is applicable. The pull–up resistor, however, should not be below 1.5kΩ. Using a very low value will increase the logic 0 voltage the master sees when reading from the network, reducing the noise immunity of the system. In difficult situations an active pull–up driver may be used to provide optimized wave forms and highest noise margins. Such a driver could also include circuitry for echo cancellation.

C. Addressable Switches

The addressable switch is the core component for creating branches of complex MicroLANs. It is a three–terminal device with a open drain output of a transistor controlled by the bus master through the 1–Wire network. When switched on, the impedance of the transistor is typically 15 Ω (DS2405, < 10 Ω with DS2407); switched off the impedance is minimum 10 MΩ. Due to its low but non–zero impedance, every switch in the path to a device in the MicroLAN creates a small voltage drop of approximately 30 mV (15 Ω x 2 mA, for example, assuming a 2.5kΩ pull–up resistor and a pull–up voltage of 5V). The more switches involved in building a path, the higher the logic 0 voltage the bus master will see when reading from the network. The addressable switch has almost no impact on the logic 1 voltage; it adds to the load of the network as any other component on the activated branches.

If a limb of a MicroLAN is deactivated by switching its addressable switch into the high impedance state, all devices on that limb and beyond loose their parasitic power. The addressable switches affected will loose their status. If deselected for approximately 1 second or greater and then powered up again, they all will have their switching transistor non–conducting. Due to the parasitic power supply, the power–up cycle will cause a short drop in the 1–Wire voltage until the energy reservoirs of the devices in the newly activated branch are filled sufficiently.

If addressable switches on branches are used to permanently (de-)activate equipment through the MicroLAN, the optoisolated interface (see previous chapter, “Building Branches”) should be used to keep the equipment running during times where the path leading to that particular switch is not powered.

D. 1–Wire Interface, Parasitic Power Supply

To understand the voltage and timing issues of the MicroLAN, one should refer to its electrical equivalent circuit (Figure 4). The MicroLAN consists of three segments, a) bus master, b) the wiring and coupling and c) the 1–Wire devices themselves.

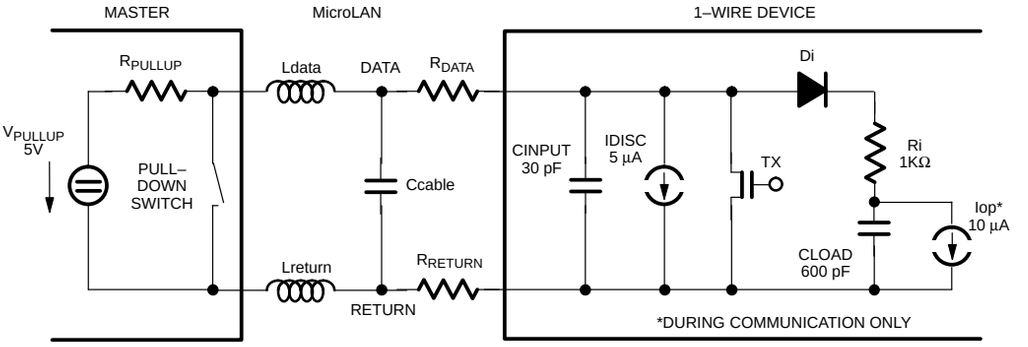
At the master’s side there is a DC voltage source representing the pull–up voltage, the single network pull–up resistor and the pull–down switch, a digitally operated transistor controlled by the bus master.

The wiring between the master and the 1–Wire devices is modeled by the inductive and ohmic resistance of the data and return lines and the lumped capacitance of the cable. The cable capacitance is calculated by multiplying cable length between the master and the most distant device by the specified cable capacitance. The inductance of importance is the differential inductance which is that measured across the cable input with the two wires of the bus shorted together at the far end. Differential inductance is substantially lower than the inductance of a single wire because the current flows in opposite directions in the pair and in an ideal case would cancel completely. The resistance of the data line represents the cable length multiplied by the specified resistance of a single wire. The resistance of the return line is the same as the resistance of the data line plus a few Ohms for each addressable switch between the bus master and the most distant branch.

A 1–Wire device is represented by its input capacitance (CINPUT), a constant discharge current (IDISC, of typically 5 μA), the parasitic power supply circuitry (Di, Ri, CLOAD) and its operating current (Iop) of 10 μA during communication. Any device residing on the network will consume operating current, even if not addressed. This current is required to keep its 1–Wire interface synchronized with the communication protocol. When conducting, its impedance is nominally 100 Ω which results in a current sink capability of 4 mA at a voltage drop of 0.4V. If multiple 1–Wire devices are residing on the bus, CINPUT, IDISC, Iop and CLOAD need to be multiplied by the number of devices; Ri needs to be divided by the number of devices. The MOSFET inside the equivalent circuit of the 1–Wire device allows the device to respond to the command by putting a logic 0 level on the network. Except for the presence detect cycle, Search ROM

command, Skip ROM and Read ROM command, only one such transistor may be conducting for short periods of a time slot when reading from an addressed device.

MicroLAN ELECTRICAL EQUIVALENT CIRCUIT Figure 4



The fanout of a MicroLAN depends on the value of the pull–up resistor and the pull–up voltage. The limit is reached as the voltage drop across the pull–up resistor reduces the MicroLAN voltage to 2.8V. This is the minimum voltage required to recharge the parasitic power supply of the 1–Wire devices. From this DC requirement, Table 1 was obtained.

1–WIRE DC FANOUT Table 1

PULL–UP RESISTOR	PULL–UP VOLTAGE		
	4.0V	5.0V	6.0V
1.5kΩ	53	98	142
1.8kΩ	44	81	119
2.2kΩ	36	67	97
2.7kΩ	30	54	79
3.3kΩ	24	44	65
3.9kΩ	21	38	55
4.7kΩ	17	31	45

As shown in Figure 4, MicroLAN devices contribute to the capacitive load on the network. While the input capacitance of the device is always present, the typical 600 pF energy storage capacitance of the internal power supply only temporarily shows up when voltage on the line is first applied and the devices start to charge their internal energy reservoirs. Its effects are also seen when the devices replenish their power supply. This happens above the logic switching level at voltage levels of 2.8V minimum and therefore does not add to the system capacitance during normal communication. On heavily loaded long lines, the effect of recharging the parasitic power reservoir can be seen as a dip on the rising edge. When activating a new branch with many 1–Wire devices, however, the communication is delayed until this precharge is satisfied.

E. UART Operation

The UART is a bi–directional, self–timed serial communication device found in most computers. After setting its communication parameters (data rate, character length, parity, number of stop bits) the UART will transmit each character that it gets from the CPU (central processing unit) according to the specified parameters. Whenever activity is found on the receive channel, that waveform is assumed to follow the same communica-

tion parameters that apply to the transmit channel. Both, the receive and the transmit channel are tied together inside the DS9097 COM port adapter to form a 1–Wire interface.

To do 1–Wire communication, the UART is set up for a data rate of 115.2 Kbps. For the UART, each time slot is equivalent to a complete character. Depending on the bit pattern of the character written to the transmit register of the UART, the result will be a Write–One or a Write –Zero time slot. The UART is in effect creating and digitizing the 1–Wire timing by choosing a particular data pattern. The Read Data time slot is essentially the same as the Write One time slot. In contrast to the Write–One time slot, however, the CPU will not discard the data it finds in the receive register after a Read Data time slot was generated. Since all data is transmitted least significant bit first, any response from the MicroLAN is found in the least significant bits of the received character.

When reading, the logic of the UART's receive channel triggers on the transition from the idle status (voltage found during times of no communication or between characters) to the active status. With every character there is a preamble, the start bit, transmitted before the data of the character itself. This helps the UART to get ready for receiving valid data. At 115.2 Kbps the duration T of the start bit as well as the data bits is 8.68 µs (the reciprocal of 115200). The highest safety margin against non–ideal communication characteristics of serial data is reached if one samples the voltage of the communication signal at the midpoint of the window that represents a bit. The best time points for sampling therefore are at 1.5T, 2.5T, 3.5T, etc., after the receive channel is triggered. For 115.2 Kbps these sampling time points are 13.02 µs, 21.70 µs, 30.38 µs, etc., after the beginning of a character or time slot, respectively.

These numbers need to be correlated to the conditions of the read data time slot, as described in data sheets. When transmitting, a 1–Wire device is guaranteed to provide the data value of a bit for a minimum of 15 µs after the beginning of a read data time slot. This means that the first sampling of the UART's receive channel at 13.02 µs occurs at a good time to catch the response of a 1–Wire device, provided that in the case of reading a logic 1 the network was able to return to the logic 1 voltage within 4.34 µs (the time from the end of the start bit until the center of the first data bit).

This condition is difficult to meet, especially if there is a long cable between bus master and 1–Wire device. Fortunately, a 1–Wire device holds the value of a bit to be transmitted for typically 30 µs rather than 15 µs, which is the absolute worst case. As a consequence, if the operating conditions of all 1–Wire devices of a MicroLAN are kept within certain limits, using the second bit the UART reads during a time slot produces more reliable results. At 21.70 µs one is still on the safe side for reading a logic 0 and there are 13.02 µs for the data line to reach a logic 1 level in case the response from the network is a logic 1. One can rely on the second bit as long as all 1–Wire devices of a MicroLAN are within the temperature range of 0°C to 50°C and the 1–Wire pull–up voltage is 5V minimum. The DS9097 COM Port adapter, for example, provides a pull–up voltage of 6V; its pull–up impedance is approximately 1.5kΩ.

The PULLUP resistor of the MicroLAN together with the cable capacitance and the input capacitances of the 1–Wire devices on the active branches of the network represent the network time constant τ. As long as the load capacitors of the parasitic power supply are not yet being recharged, this network time constant determines the speed at which the MicroLAN data line returns to a logic 1 voltage. This waveform can be calculated as

$$V(t) = V_{PULLUP} * (1–\exp(–t/\tau))$$

With the requirement that at t = 13.02 µs the 1–Wire voltage needs to have reached the 2.2V threshold of a logic 1, the value of τ can be calculated for any valid PULLUP voltage as

$$\tau = 13.02\ \mu s / \ln(V_{PUP}/(V_{PUP}–2.2V))$$

The results are as shown in the table below:

MAX. NETWORK TIME CONSTANT (13.02 µs SAMPLING)	PULL–UP VOLTAGE		
	4.0V	5.0V	6.0V
τ	16.3 µs	22.4 µs	28.5 µs

Assuming the MicroLAN is loaded with the maximum number of devices as shown in Table 1, the cable capacitance alone must not exceed the value in Table 2 to yield the network time constant shown above.

MAXIMUM CABLE CAPACITANCE AT MAXIMUM NUMBER OF DEVICES (13.02 µs SAMPLING) Table 2

PULL–UP RESISTOR	PULL–UP VOLTAGE		
	4.0V	5.0V	6.0V
1.5kΩ	9.27 nF	12.00 nF	14.73 nF
1.8kΩ	7.72 nF	10.00 nF	12.28 nF
2.2kΩ	6.32 nF	8.18 nF	10.05 nF
2.7kΩ	5.15 nF	6.67 nF	8.19 nF
3.3kΩ	4.21 nF	5.46 nF	6.70 nF
3.9kΩ	3.56 nF	4.62 nF	5.67 nF
4.7kΩ	2.96 nF	3.83 nF	4.70 nF

Assuming a typical specific cable capacitance of 50 pF/m, Table 2 translates into the cable lengths shown in Table 3. The length values can easily be scaled to other cable capacitances by multiplication with the actual specific cable capacitance divided by 50 pF/m.

MAXIMUM CABLE LENGTH AT 50 pF/m (13.02 µs SAMPLING) Table 3

PULL–UP RESISTOR	PULL–UP VOLTAGE		
	4.0V	5.0V	6.0V
1.5kΩ	185 m	240 m	295 m
1.8kΩ	154 m	200 m	246 m
2.2kΩ	126 m	164 m	201 m
2.7kΩ	103 m	133 m	164 m
3.3kΩ	84 m	109 m	134 m
3.9kΩ	71 m	92 m	113 m
4.7kΩ	59 m	77 m	94 m

If using less than the maximum possible number of devices, the maximum cable capacitance as shown in Table 2 may be increased by 30 pF for each device less than the maximum number shown in Table 1.

So far, the UART description was not specific in the number of bits per character for the generation of 1–Wire time slots. Using six bits per character plus start bit, a character data pattern of all zeros would result in a logic 0 level for 7 x 8.68 µs, which is 60.76 µs. This matches the descriptions of the Write Zero Time Slot as specified in data sheets. A reasonable Write One Time Slot is achieved if the data pattern of the character is all ones; in this case the resulting wave form would have a logic 0 level for 8.68 µs caused by the start bit. Between characters there is an idle time inserted by the UART, called stop bit. If not otherwise specified, this stop bit has the same duration as the start bit, which is 8.68 µs at a data rate of 115.2 Kbps. All together, six data bits, one start bit and one stop bit result in a total length of 8 x 8.68 µs or 69.4 µs for a time slot including an idle time for recovery of 8.68 µs. These 69.4 µs are equivalent to a data rate of 14.4 Kbps, which is the fastest 1–Wire data rate possible through a UART, assuming that no time is

required by the microprocessor for reading and reloading the UART to provide the next character pattern.

In case of a network with a capacitive load close to the maximum possible value, the idle time of 8.68 µs following a write zero time slot is too short for the voltage on the data line to reach the minimum of 2.2V for the 1–Wire devices to recognize the end of a time slot. To increase the idle time, one can increase the character length by 1 or 2 bits. These additional bits need to be set to logic 1 to achieve the desired effect. Otherwise they would add to the time the data line stays at logic 0 level during a write zero time slot. The possible options and their practical results are shown in the table below.

IDLE TIME	8.68 µs	17.36 µs	26.04 µs
CHAR. LENGTH	6 BITS	7 BITS	8 BITS
WRITE ZERO PATTERN	00 0000	100 0000	1100 0000
DATA RATE	14.4 kbps	12.8 kbps	11.5 kbps

As long as the network time constant does not exceed 30% of the maximum allowed value for a given pull–up voltage, the shortest idle time is sufficient. For heavier loaded MicroLANs, the 17.36 µs idle time or more may be required. In any case, before building a huge MicroLAN it is highly recommended to check the operating conditions as explained in the section III.D., Electrical Verification.

As the table shows, longer idle times reduce the possible data rate of the MicroLAN. A lower data rate increases the response time of the network for detecting changes in topology and/or population. At 14.4 Kbps the ROM search will discover 66 devices per second, not taking into account the limited speed of the bus master itself and the time to select the switches and precharge the limbs.

F. Computer Speed and Operating System

The timing on the MicroLAN is not solely determined by the UART and the network characteristics. The hardware design of the bus master itself and the software running on it also have impact on the performance of the network. Among other things, the task of the master is to read data from the UART and to reload it with the character for the next time slot. When transferring data exclusively (read or write many bytes) the behavior of the master is very predictable; it adds a constant delay between the characters read from or written to the UART. In case of a Search ROM procedure, the master has to do some extra data processing to determine the value of the next character before loading it into the UART's transmit register. This adds a varying delay between the characters, potentially extending the idle time between time slots significantly with respect to transferring data only. In any case, the delay depends on the clock frequency applied to the microprocessor, the size of the program cache memory, the design of the application software and the operating system.

If the program loop being executed for reading or writing a stream of data fits into the fast program cache memory, the delay will be very short since no access to the slow main memory will be required to load the next block of program code. A sophisticated operating system burdens the bus master with some overhead that needs to be done at regular intervals, thereby slowing down servicing the UART.

Some very recent computer designs try to compensate for this delay by using a UART with FIFOs in the receive and transmit channel. Unfortunately, for 1–Wire communication the master often needs to know the value of the most recent bit before being able to provide the next bit, this approach will not work for MicroLAN applications. Therefore, if one is going to use such a computer as bus master for a MicroLAN, the FIFOs need to be disabled. This can be done under software control.

The delay between characters or time slots introduced by the bus master is difficult to predict. Measurements have been made using a 80386DX–based IBM–compatible computer running at 33 MHz. The computer had to measure the total time it needed to access a DS1993 and read the entire data memory. The UART was setup for a character length of 8–bits. Under DOS the computer–introduced delay was 20 µs; the Windows–result was 30 µs. For an 80486–based computer at 33 MHz clock rate one might expect half of this delay. These numbers cannot be scaled for other clock rates; if

required they need to be measured in a test setup. Although the computer–introduced delay slows down the communication speed of the MicroLAN, it gives the 1–Wire devices another opportunity to recharge their parasitic power supply.

III. MicroLAN OPTIMIZATION

A. COM Port to MicroLAN Adapter

For most MicroLANs the DS9097 COM port adapter is sufficient. For heavily loaded lines, extreme temperatures or maximum speed a more sophisticated adapter is required. Such an adapter consists of an RS232 level translator, a MicroLAN driver and a passive pull–up as outlined in the functional block diagram of Figure 5. The driver circuit includes an active pullup and a slew–rate controlled pull–down transistor that translates the TXD signal from the COM port into 1–Wire time slots.

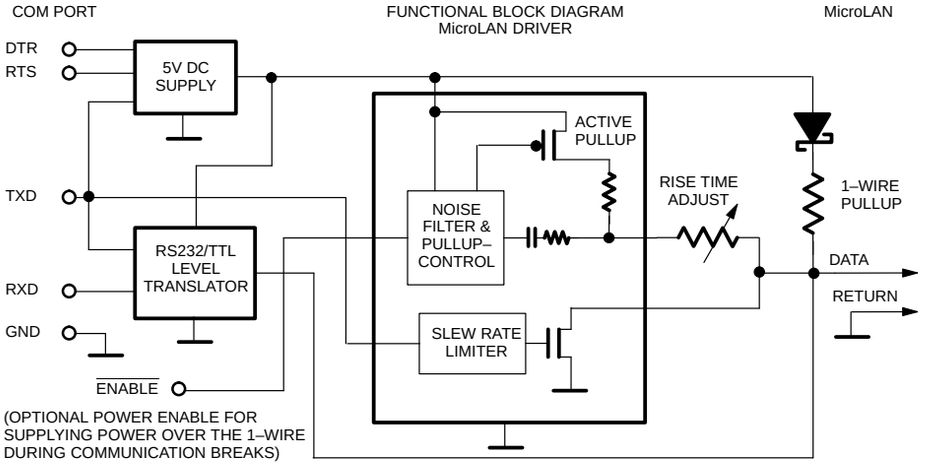
The value of the 1–Wire pull–up resistor may range from 1.5 kΩ to 4.7 kΩ, as explained in chapter II. The purpose of the diode in series with the pull–up resistor is to prevent energy flow into the circuit if the interface is not powered up. In the unpowered state the interface should have a high impedance to allow another bus master to be connected. Due to its low 0.2V forward voltage drop a Schottky diode is preferred over a standard silicon diode.

In Figure 5 the supply energy for the interface is taken from the RS232 control lines DTR and RTS as well as from the transmit data line TXD. This approach works reliably if the COM port drivers meet the RS232C standards. The level translator directly connects to the MicroLAN and converts its 0V/5V levels to +12V/–12V RS232 levels. Its negative voltage supply is derived from TXD during idle times and read data time slots.

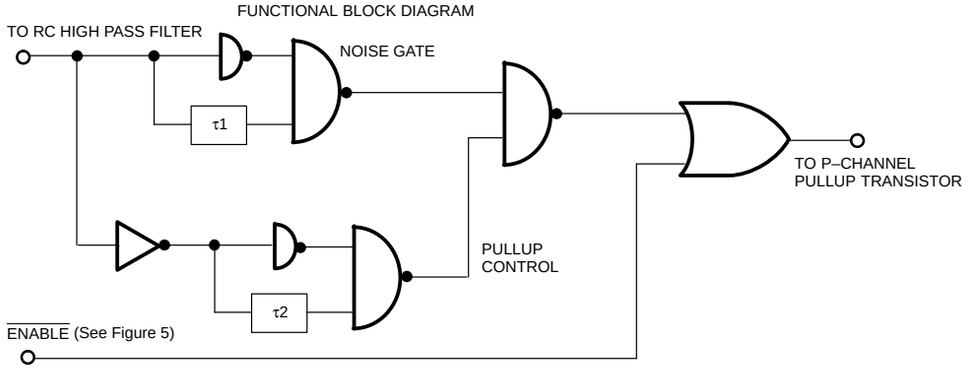
Since on longer cables there could be reflections causing positive edges, the active pull–up must be made immune to such noise during the time reflections can occur. For this reason the Noise Filter and Pull–up Control is found as part of the MicroLAN driver block in Figure 5. The control circuit is coupled to the MicroLAN via an RC high pass filter.

How the noise filter and pull–up–control circuit works is shown in Figure 6. It consists of two parallel paths feeding into a NAND gate that directly controls the pull–up transistor. The upper path of the control circuit is basically a slope detector that generates a HIGH pulse of duration $\tau 1$ at the node Noise Gate every time the MicroLAN is pulled LOW by the master. The duration of $\tau 1$ is nominally 7.5 µs. The lower path of Figure 6 is very similar to the upper path. The principle difference is the additional analog inverter in front of the slope detector. Inverter and slope detector together generate a HIGH pulse of duration $\tau 2$ at the node “Pull–up Control” every time the MicroLAN goes back to a HIGH level when it is released from a LOW. The gain of the analog amplifier determines at which voltage level the Pull–up Control signal becomes active. The duration of $\tau 2$ is nominally 7.5 µs. The NAND gate finally lets the Pull–up control signal pass to the pull–up transistor if the noise gate is inactive. This prevents the active pull–up from being activated by reflections that can occur during the first few microseconds after the line is pulled LOW. Depending on the cable length it may be necessary to limit the active pull–up current by adjusting the resistor between the pull–up transistor and the MicroLAN (Rise time adjust).

OPTIMIZED COM PORT TO MicroLAN ADAPTER Figure 5



NOISE FILTER AND PULLUP CONTROL CIRCUIT Figure 6



B. MicroLAN Topology

For best performance of a MicroLAN installation one has to find the conditions under which the loading of the MicroLAN is minimized for a given number of nodes. Assuming a topology where the number of branches from each branch level is constant, the optimum number of branch levels can be calculated.

To explain how this optimization is done one has to define a few parameters:

M = Number of target devices (not counting address able switches) to be reached via the feeder net–work.

N = Number of branch levels (limb is first level, twig second level, stem is third level, etc.)

S = Number of branches including leaves (not counting markers) branching off/connected to each level.

L = The total load, defined as the number of listening MicroLAN devices when a target device is addressed.

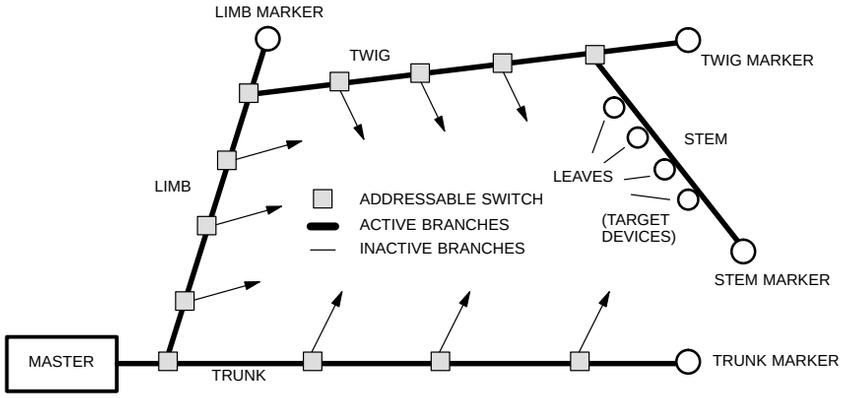
It can be shown theoretically that the optimum value of branches on each branch level is approximately 3.6 and that this value does not depend on M. This result applies to a configuration where each branch has one marker in addition to the addressable switches. The target devices are located on the most distant branches. Since the non–integer value of 3.6 cannot be realized in practice, for the following example the value of four has been chosen. The Table 4 shows how the capacity M and the load increases with each branch level. It also proves that even with more than 4096 target devices the DC fanout is not the limiting factor for the size of the MicroLAN.

Figure 7 shows a MicroLAN example of four branch levels according to this table. The number of listening devices is found by counting the addressable switches and markers on the active branches and the leaves (= target devices) on the most distant branch (total of 20). Since there are four leaves per stem, four stems per twig, four twigs per limb and four limbs at the trunk, the number of target devices (not counting addressable switches and markers) is 4 x 4 x 4 x 4 = 256 without having more than 20 devices listening at a time.

MINIMUM LOAD TOPOLOGY Table 4

N (NUMBER OF BRANCH LEVELS)	L (LOAD OF LISTENING DEVICES)	M (TARGET DEVICES)
1 (TRUNK ONLY)	5	4
2 (TRUNK, LIMBS)	10	16
3 (TRUNK, LIMBS, TWIGS)	15	64
4 (TRUNK, LIMBS, TWIGS, STEMS)	20	256
5	25	1024
6	30	4096

MINIMUM LOAD TOPOLOGY Figure 7



C. Protection and Noise

When interfacing a MicroLAN to a computer one has to provide a means to protect the system from damage by ESD (electrostatic discharge) and EMI (electromagnetic interference) from radio stations, communication equipment and lightning, etc. All MicroLAN compatible 1–Wire devices have built–in ESD protection circuitry to withstand ESD events (±10kV minimum Human Body Model). At the bus master's end of the MicroLAN, reliable protection can be provided by means of diodes shorting negative spikes or feeding positive spikes higher than 5V to the power supply unit where they are smoothed by capacitors and other protection devices. When selecting protection diodes or surge suppresses to be connected to the data line one should choose devices with minimum junction capacitance and fast switching times.

The cable recommended for MicroLANs is regular unshielded twisted pair category 5 cable. Such cable is available with two or more pairs of wires. The fact that the wires are twisted reduces unwanted coupling from nearby power cables or analog phone lines by canceling the induced voltages. The specific capacitance between the wires of a pair is approximately 50 pF/m; between wires of different pairs a value close to 30 pF/m

can be expected. Unused wires need to be left unconnected at both ends of the cable. Grounding them is worse than leaving them floating: it can increase the capacitive load on the MicroLAN so significantly that communication becomes impossible. It is also not recommended to run two MicroLANs through the same cable, unless each pair is separately shielded, and the cable also shielded. The recommended shielded cable is IEEE 1394 “Firewire”.

D. Electrical Verification

After the topology of a MicroLAN application has been defined it is recommended to check the electrical feasibility. This is done using the equivalent electrical circuit of Figure 4. The total capacitance of the cable is lumped into Ccable. RDATA is the impedance of the data line proportional to the cable length. R_{RETURN} represents the impedances of all addressable switches in the branch plus the wire impedance of the return line (same length as the data line). The diode is considered to have a voltage drop of 0.7V and be ideal otherwise. For simplicity, the effect of Ldata and Lreturn is not included in this calculation.

To verify the electrical feasibility of a MicroLAN perform the following steps:

1. calculate L1, the maximum cable length when activating the most distant branch
2. calculate L2, the cable length when activating the most heavily loaded branch (branch with most target devices)
3. count N1, the number of devices loading the MicroLAN in case of the longest cable
4. count N2, the maximum number of devices loading the MicroLAN in case of the heaviest load
5. calculate the total capacitances;
   Ccable1 = L1 x ccable
   Ccable2 = L2 x ccable
   C1 = Ccable1 + N1 x CINPUT
   C2 = Ccable2 + N2 x CINPUT
   Ctotal is the higher value of C1 and C2.
   ccable is the specific cable capacitance.
6. Using N2, select the pull–up resistor for the available pull–up voltage from Table 1. If possible select a pull–up resistor even lower than the table indicates, but not less than 1.5kΩ.

   If the load is too high for the available pull–up voltage, split up the most heavily loaded branch to reduce the DC load and go back to step 1.
7. calculate the network time constant by multiplying Ctotal x R_{PULLUP}.

   The passive pull–up may be sufficient if the network time constant does not exceed 16.3 µs at 4V pull–up voltage, 22.4 µs at 5V or 28.5 µs at 6V, respectively. If the time constant is higher, reduce Rpullup as far as possible. If this is not possible or sufficient, an active pull–up driver is required.
8. calculate the maximum LOW voltage at the master’s end of the MicroLAN
   $V_{ml} = V_{PULLUP} \times (R_{DATA} + R_{RETURN}) / (R_{DATA} + R_{RETURN} + R_{PULLUP})$.
   The result needs to be less than 0.8V; otherwise the bus master might have difficulties in recognizing a logic 0 level on the MicroLAN.
9. check the recharge condition, i.e. verify that between time slots there is sufficient idle time for the MicroLAN to allow recharging the parasitic power supply of its 1–Wire devices.

With a parasitic power supply capacitor CLOAD = 600 pF and a current demand of 10 µA the capacitor will lose 1.0V during a write 0 time slot. To be able to tolerate an indefinite series of Write 0 Time slots, the idle time between time slots needs to be long enough to restore the energy consumed previously. To calculate the recharge time it is assumed that the operating current stops and recharging CLOAD starts as soon as 2.8V are reached on the data line of the MicroLAN. The discharge current Idisc will continue flowing. As soon as CLOAD is being recharged the network time constant changes from

$\tau 1 = R_{PULLUP} \times (Ccable + N \times CINPUT)$ to
$\tau 2 = R_{PULLUP} \times (Ccable + N \times (CINPUT + CLOAD))$.

At a load of 20 1–Wire devices , for example, this easily doubles the value of the time constant, even if cable was already a high capacitive load.

The effective voltage available on the MicroLAN to recharge the parasitic power supply will be

$$\Delta U = V_{PULLUP} – 2.8V – N \times R_{PULLUP} \times IDISC.$$

Since 1.0V was lost during the write 0 time slot, the voltage on the MicroLAN needs to rise by 1V at a time constant of $\tau 2$ and a voltage of ΔU available for recharging. Solving the differential equation the required recharge time is:

$$trecharge = \tau 2 \times \ln (\Delta U / (\Delta U – 1V)).$$

This recharge time begins as the voltage on the data line has reached 2.8V. With a network that reaches the 2.2V level at 13 µs after the bus master has stopped pulling the data line to 0 V, the 2.8V level will be reached after no more than 20 µs maximum. The idle time between time slots therefore needs to be:

$$tidle = 20\ \mu s + trecharge.$$

The recharge condition is fulfilled if the idle time generated by the UART plus the inter character delay introduced by the bus master is greater or equal to the required tidle as calculated above. For example, a UART that is setup for 8–bits character format provides an idle time of 26 µs. If the bus master is not one of the fastest models, it may add 20 µs to the idle time between time slots. In this case the recharge condition is fulfilled if the recharge time does not exceed 26 µs.

If the recharge condition is not fulfilled with the R_{PULLUP} and V_{PULLUP} chosen, R_{PULLUP} needs to be reduced further (not below 1.5kΩ) or V_{PULLUP} needs to be raised (up to 6.0V), if possible. If this is not sufficient to fulfill the recharge condition, the network topology needs to be changed to reduce the load from the devices on the network or an active pull–up driver as outlined in Figure 5 will be required.

Checking the recharge condition in case of a 4V pull–up voltage, one might find very long recharge times. With a high number of devices it might be impossible to calculate the recharge time since the maximum voltage for recharging is less than the 1.0V required to compensate for the energy loss. Nevertheless, the network might work if the idle time is very long, e. g., in the range on 100 µs or more. The reason for this comes from the fact that IDISC is not as constant as assumed for the calculation. It may be in the range of 1.5 to 3 µA if the voltage on the data line is very low. Loss of energy at the parasitic power supply will reduce the threshold voltage where the recharging starts to a value close to 2.3V. This together with a lower value of IDISC will allow a slow MicroLAN operation even at 4.0V pull–up voltage. However, for reliable operation at a reasonable speed one should avoid such a low pull–up voltage. Whenever possible, the pull–up voltage should be in the range of 5.0 to 6.0V.

Sample Calculation

(The cable inductance is not included in this calculation.)

rcable = 0.085 Ω/m
ccable = 30 pF/m
V_{PULLUP} = 5V
L1 = 220 m
L2 = 250 m
N1 = 40
N2 = 45
C1 = 6.6 nF + 1.2 nF = 7.8 nF
C2 = 7.5 nF + 1.35 nF = 8.85 nF
Ctotal = 8.85 nF
R_{PULLUP} = 2.7kΩ (taken from fanout Table 1)
Network Time Constant = 2.7kΩ x 8.85 nF = 23.9 µs

This is too long for 5V pull–up voltage. R_{PULLUP} therefore needs to be reduced to 2.2kΩ. This results in a network time constant of 19.5 µs, which is sufficient for the data line to reach a logic 1 value when sampling at 2.5T or 21.7 µs after the beginning of a time slot.

R_{DATA} = 250m x 0.085 Ω/m = 21 Ω
maximum 3 addressable switches (15 Ω each) required to reach the target branch.

R_{RETURN} = Rdata + 3 x 15 Ω = 66 Ω
Vml = 0.19 V.
$\tau 2$ = 97.0 µs
ΔU = 1.59 V
trecharge = 96 µs

This value is not acceptable. R_{PULLUP} is now reduced to 1.5 kΩ. These are the new results:
$\tau 2$ = 53.9 µs
ΔU = 1.86 V
trecharge = 41.6 µs

This value is better, but still very long. The pull–up voltage is now raised to 6.0V. This results in:
$\tau 2$ = 53.9 µs
ΔU = 2.86 V
trecharge = 23.2 µs

This value is reasonably short. Since the pull–up resistor and pull–up voltage were changed, the new maximum LOW voltage needs to be checked.
Vml = 0.33 V

This value is within the allowed range. The MicroLAN in this example will work if the UART is setup for 8–bits (yielding 26 µs idle time) and the bus master itself is slow enough to add 17 µs minimum to each time slot.

IV. SUMMARY

A heavily loaded large MicroLAN implementation works most reliably if the following conditions are met:

- controlled slew rate of about 3 ... 4 µs
- minimum cable capacitance and inductance (Category 5 twisted pair or IEEE 1394 “Firewire”)
- line capacitance completely discharged in every time slot
- reverse–biased Schottky diode termination at the far end of the cable
- active pull–up driver circuit (Figure 5)
- late sampling at 2.5T or 21.7 µs respectively
- maximum 1–Wire pull–up voltage
- multi–level branching topology (Figure 7)

A more detailed discussion of the most important of these recommendations is found in Application Note 108, “MicroLAN – In The Long Run”.