LON AND LAN NETWORKS CONNECTION BASED ON NEURON CHIP AND ATMEL MICROPROCESSORS

Hubert Wojtowicz **) Marcin Skuba **) Wieslaw Wajs *)

University of Mining and Metallurgy, Institute of Automatics, Krakow, Poland *)

Krosno Technical University, Poland **)

Abstract: A hardware and software for microprocessor external digital circuits using programmable device is proposed. Each node of the Local Operating Network (LON) is independently working device, and except network communication port for LON, contains eleven I/O ports that can be connected to the Local Area Network (LAN). The paper describes how to build a host interface to the NEURON CHIP microprocessor that uses the Microprocessor Interface Program. The Microprocessor Interface Program establishes a fast direct link from the host processor to the NEURON CHIP network processor. The NEURON CHIP parallel I/O object permits bi-directional data transfer to the node that is realized ATMEL based microprocessor.

Keywords: Programmable Devices, Distributed Control System, Local Operating Network, Local Area Network, Digital Circuits

1. INTRODUCTION

The NEURONCHIP® is a product of Motorola (LONWORKS Technology Device Data, 1997) or Toshiba Corporation (NEURONCHIP TMPN31503120, 1995). ATMEL Corporation is famous microprocessor manufacturer. The LON(TM) technology coming from ECHELON Corporation. ECHELON creates the network channels technology for Local Operating Network architecture (LONBUILDER MIP, 1995). The architecture of networks, that is considered in the paper, contains several elements (see Fig.1).

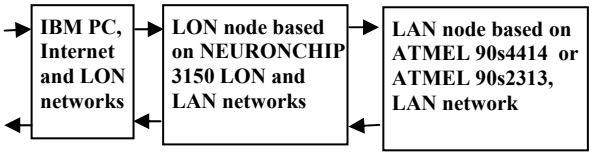

Fig 1. LON and LAN networks structure

Microprocessor Motorola 3150 is the central processor unit of the node that is called "NEURON CHIP". "NEURON C" is a programming language for this CHIP, and it is based on the ANSI C language.

The node NEURON CHIP based microprocessor can be used to decode asynchronous serial data coming into the node. The serial I/O object (see Fig. 2) can be used to decode an asynchronous serial data stream into 8-bit characters with start and stop bits.

2. PARALLEL I/O INTERFACE

The physical interface to the parallel I/O object is realised through the eleven I/O pins of the NEURON CHIP microprocessor. No other I/O objects of the NEURON CHIP may be used for parallel I/O (see Fig. 2).

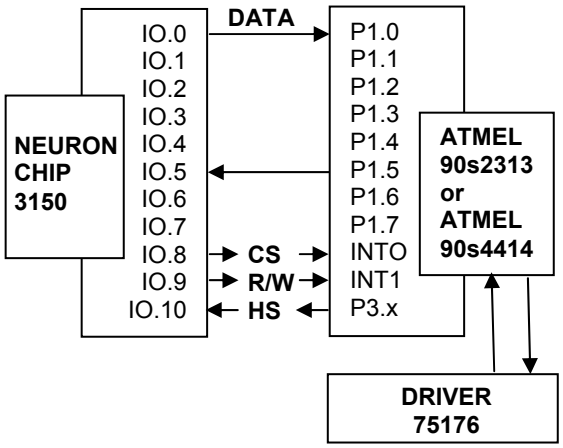

Fig 2. Pins assignments for LAN up to 255 nodes

The NEURON C programming language provides several build in functions that enable the use of the parallel I/O object. We can not considered the hardware level knowledge of the handshaking protocol (LONBUILDER MIP, 1995). The NEURON CHIP based node provides several modes of the operation for the parallel I/O object.

One of the important restrictions is the quantity of I/O pins. We can typically program eleven lines that can be programmed either as inputs or as outputs (LONWORKS Technology Device Data, 1997). Limited number of I/O lines makes it impossible to connect too many sensors and actuators directly to a NEURON CHIP-based node. It is possible to build Data Bus that is working as the extension of the node's I/O port with external devices connected into it. Each of these devices is based on ATMEL 90s4414 or ATMEL 90s2313. A device is responsible for two main tasks: bus service i.e. the task for the data exchange, and tasks addressed for each device that is connected to the NEURON CHIP based node.

3. NEURON CHIP AND ATMEL MICROPROCESSOR IN MASTER-SLAVE MODE

The NEURON CHIP parallel I/O object permits bi-directional data transfer at rates of up to 3.3 Mbps (LONWORKS Technology Device Data, 1997). A NEURON CHIP can communicate with ATMEL microprocessor. The Microprocessor Interface Program (Wajs, 1998b and 2000) running on the NEURON CHIP provides an easy connection from NEURON CHIP based node to ATMEL based node (see Fig. 3) [6].

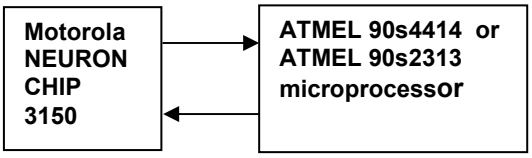

Fig 3. Master-Slave connection for the ATMEL and NEURON CHIP 3150 microprocessor

Each line of node's I/O port can be declared as one-way line. We can set four lines as data input lines, and also four lines as data output ones. During one cycle of transmission (sending or receiving the message) we can transmit half byte (nibble). Data lines transmission is supported by three following control lines: R/W (Read/Write) – it is driven by node based on NEURON CHIP microprocessor and determines data flow (see Fig. 2). If the state of (R/W) line is low then node sends the data. If it is in high state then node reads data transmitted from the external device. The CS (Chip Select) line is used to verify that the valid data has been sent to the Bus, or the node has already read the data. This line is also driven by node based on NEURON CHIP microprocessor. The HS (Handshake) line also confirms the read or write data status, as CS line described above, but it is driven by ATMEL microprocessor. An additional line is not connected to the node based on NEURON CHIP. It is the busy line used to synchronize nodes based on ATMEL microprocessor during simultaneous data reading. Transmission begins if node drives R/W signal to low. This change causes external interrupt signal to appear on each of Bus devices (INT1). All devices respond at the same time because interrupt priority is set to the highest of all possible values. First byte of the packet sending is received by all devices, because it is an address byte, and then it is compared with the internal device's address. If it doesn't match then the device leaves communication procedures, and waits for next packet. If it matches (we assume that there is only one such device address in the network because each node based on ATMEL microprocessor has its own unique address) device reads the packet and then sends its response (also packet of the data) to the node. The structure of sending packet is divided into three parts. The head consists of two bytes: address and packet length. Bits of the address byte have following meaning: two most significant bits specify the place where device is located into the network, next four bits describe device type, and if we have more than one device then two less significant bits specify the device number. The body of the packet contains: requests, commands, and the data that is sending either by the node based on NEURON CHIP or by the node based on ATMEL microprocessor. The control part containing check sum of the packet. Each node has a list of devices that are present in LAN. The node takes addresses of devices from the list, and sends packet of the data with commands to desired device. If transmission is finished node takes address to the next device and communication process starts again. If communication process to the node failed then it is repeated several times. At the end of the communication process repetition the node sends alert. If timeout is exceeded the node breaks communication process, waits for some time, and then starts it again. There is no possibility of system suspension.

In a master-slave configuration, the master drives IO8 as a pin chip select and IO9 pin to specify a read or write cycle, and the slave drives IO10 as a handshake acknowledgement (see Fig. 2). The data transfer rate scales proportionally to the input clock rate. After every byte read or byte write the handshake line is monitored by the master to verify if the slave has completed processing (when HS=0), and then the slave is ready for next byte transfer. Handshaking allows the master to monitor the slave between every byte transfer, ensuring that both processors are ready for the byte to be transferred. This is done automatically in NEURON CHIP to ATMEL data transfer. When configuring in master-slave mode, the ATMEL accepts IO8 as a Chip Select (CS) and IO9 to specify whether the master will read or write (R/W).

When CS is asserted and either IO10 is low (0) or IO10 is high (1) and R/W is low (0), pins IO0 trough IO7 read form the bi-directional Data Bus. We have (R/W=0 and IO10=1) or ((R/W=0 or 1) and IO10=0).
When IO10 is high (1), R/W is high (1), and CS is asserted, IO10 is driven as the handshake (HS) acknowledgement signal to the master. We have (R/W=1 and IO10=1).

4. AN EXAMPLE OF PARALLEL I/O INTERFACE

A NEURON CHIP may communicate with any other microprocessor. The Microprocessor Interface Program running on the NEURON CHIP provides an easy solution for connecting the NEURON CHIP to a microprocessor.

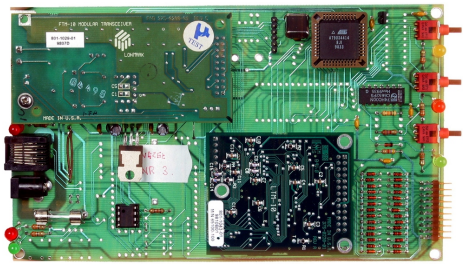

Fig 4. The NEURON CHIP microprocessor node connects LON and LAN networks

The physical interface to the parallel I/O object is realised through the eleven I/O pins of the NEURON CHIP microprocessor.

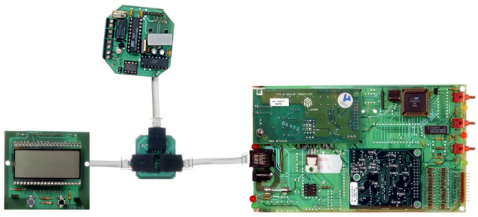

Fig 5. LAN based on ATMEL microprocessor is connected to the LON node based on NEURON CHIP microprocessor

Since, the number of devices is very limited it is possible to build Data Bus that is working as an extension of the node's I/O port with external devices connected into it (see Fig. 5). Each of these devices is based on ATMEL microprocessor [5]. The External Device is responsible for two main things: Bus service i.e. procedures for data exchange, and jobs for each of devices. We assume that the external device is an ATMAL based node. The possible number of nodes based on ATMEL microprocessor is less than 256 in one LAN connected to the node which has a Neuron ID (Neuron ID it is a unique 48 bits number given by manufacturer). Each line of node's I/O port can be declared as one-way line, and we can set four lines as data input lines, and also four lines as data output lines. Transmission begins when node drives R/W signal into low. This change causes the external interrupt signal to appear on each of Bus devices (INT1). All devices respond at the same time because the interrupt priority is set to the highest of all possible values.
Each node has a list of devices that are present in the system. It takes an address of device from this list and sends packet of data with commands and requests to the desired device. When transmission is finished node takes address of next device and communication starts again.

5. SOFTWARE FOR LAN PROTOCOL

Standard Network Variable Types (SNVT's) facilitate interoperability by providing well-defined interface for communication between nodes made by different manufactures. A node may be installed in a network and logically connected to other nodes via network variables as long as the data types match. The representation for all fixed point numeric SNVT's is as follows:

SNVT Range	***Type Definition***
0 … 65535	unsigned long
-32768 ... 32767	signed long
0 … 255	unsigned short
-128 … 127	signed short

The representation for the floating point SNVT's is ANSI/IEEE 754 floating point: 1 sign bit, 8 exponent bits, and 23 mantissa bits, for a total of 32 bits. For all the floating point SNVT's, the range is approximately from –1E38 to +1E38 units. Floating point objects may be declared as local variables and as network variables, and can be communicated across the network. These data types are compatible with the NEURON C Extended Arithmetic type float_type. SNVT's are identified as structures. A definition of each SNVT structure is provided below and information on specific fields is summarized in Table.

The structure definition used by **NVCFG**

```
Typedef  struct {
   Unsigned int Counter;        //Packet number
   Unsigned int  Type;          //Packet type
   Unsigned int  Len;           //Data length for array Data
   Unsigned int Data[MAX_NVDATA];
      //Packet data, array set up for max length of data
   Unsigned int CheckSum;    //Check sum
} NVCFG;
```

The structure definition used by **NVDEV**

```
Typedef  struct {
   Unsigned int Counter;        //Packet number
   Unsigned int UID;            //Device identifier
   Unsigned int  DType;         //Device type
   Unsigned int Req;            //Command type
   Unsigned int  Len;           //Data length for array Data
```

```
    Unsigned int Data[MAX_NVDATA];
      //Packet data, array set up for max length of data
    Unsigned int CheckSum;    //Check sum
} NVDEV;
```

A value of network variable type NVDEV and NVCFG can be saved into MS Access local data base. There are four arrays where values are saved. The table ReqCfg is a collection of the data that we need for the LON configuration description. The structure of the Table ReqCfg:

The structure of the Table ReqCfg:

Field identifier	Field Type	Description
ID	long	Unique record identifier
IdPacket	int	Packet number into the sequence for node (0-255)
TypePacket	int	Packet type (1-4)
DIArray	int	Length of Array
T1	int	first element of array
T2	int	second element of array
T3	int	third element of array
Time	Date, hour	the time at which the packet is sent
Node	Text	the name of the node that the packet is addressed
Received	True/False	True - if the record is received by the server LON

The structure of the Table RespCfg:

Field identifier	Field Type	Description
ID	long	Unique record identifier
IdPacket	int	Packet number into the sequence for node (0-255)
TypePacket	int	Packet type (1-4)
DIArray	int	Length of Array
T1	int	first element of array
T2	int	second element of array
T3	int	third element of array
Time	Date, hour	the time at which the packet is sent
IP	Text	IP for the local computer

The structure if the Table ReqDev:

Field identifier	Field Type	Description
ID	long	Unique record identifier
IdPacket	int	Packet number into the sequence for node (0-255)
IdDev	int	Device identifier
TypeDev	int	Device type
TypePacket	int	Packet type (1-4)
DIArray	int	Length of Array
T1	int	first element of array
T2	int	second element of array
T3	int	third element of array
Time	Date, hour	the time at which the packet is sent
Node	Text	the name of the node that the packet is addressed
Received	True/False	True - if the record is received

The structure of the Table RespDev:

Field identifier	Field Type	Description
ID	long	Unique record identifier
IdPacket	int	Packet number into the sequence for node (0-255)
IdDev	int	Device identifier
TypeDev	int	Device type
TypePacket	int	Packet type (1-4)
DIArray	int	Length of Array
T1	int	first element of array
T2	int	second element of array
T3	int	third element of array
Time	Date, hour	the time at which the packet is sent
IP	Text	IP for the local computer
Received	True/False	True - if the record is received

6. CONCLUDING REMARKS

The solution that is proposed can be a low cost automation system. It can be competitive for the end user. A central heating control system (Wajs, 1998a and Wajs, 1998c) is the first one where the algorithm is working. The thermal comfort can be reached independently for every room. The temperature sensors and valves at heaters are installed so we need network external devices – LAN sensor node and LAN actuator node for Distributed Control System (DCS). Every user has ability to set his own desired temperature value. The node gets the temperature value from sensor device, and by control algorithm determines the position of the valve at the heater.

We realised more complex algorithm for central heating systems – algorithm optimising energy consumption. Signals from sensors can be used for this purpose, especially in office rooms. The algorithm based on Matlab Neural Network toolbox allows more efficient control of energy consumption. A user of the system can set up the set point arbitrarily. In spite of this fact the system can calculate the optimal value of the set point for each point of the Distributed Control System. The user can decide whether his system should work as an optimally working control system, or as a system that reflects personal preferences of the user.

It is possible to use features of the LONWORKS technology. The DDE Server mechanism allows for exchange of the data among nodes and PC, and also among many different computers via Internet. It allows for a remote control in an intuitive and simple manner. A security system is another place where that system is working. There are sensors detecting movement in each of the rooms. External devices are continuously checking state of the sensors. The device is ready to send an alert to every point of the DCS.

REFERENCES

LONWORKS, Technology Device Data, Rev. 4, DL 159/D, Q4/97, Motorola Austin, Texas.

LONBUILDER Microprocessor Interface Program (MIP) User Guide, LONWORKS Engineering Bulletin, Echelon Corporation 29500/078-0017-01, Palo Alto (1995).

NEURONCHIP TMPN3150/3120. Data Book, 5205B, Toshiba Corporation 1995.

Parallel I/O Interface to the NEURON CHIP LONWORKS Engineering Bulletin, Echelon Corporation, Palo Alto (1995).

Wajs W., "Integracja Systemu LUMEL Ciepło z Siecią LONWORKS" PAK, Warszawa, no 12 pp. 461-463, 1998a (in polish).

Wajs W., „Introduction to the LONWORKS Technology" in „Integrated Control Systems and Intelligence Control" TEMPUS JOINT EUROPEAN PROJECT S-JEP-11317-96 Krakow, 1998b.

Wajs W., A Method for Microprocessor External Digital Circuits using Programmable Devices. Proc. of IFAC Workshop PDS 2000, pp 59-61, TU Ostrava, 2000.

Wajs W., „Zastosowanie przetworników KFAP S.A. w systemach opartych na technologii LONWORKS" PAK, Warszawa, no.11 pp. 426-428, 1998c (in polish).