THIS SPEC IS OBSOLETE

Spec No: 001-25599

Spec Title: Intelligent Sensor Network - Physical and Data Layers - AN2346

Sunset Owner: Andriy Smetana (ANDI)

Replaced by: None

Intelligent Sensor Network – Physical and Data Layers

AN2346

Author: Andrew Smetana
Associated Project: Yes
Software Version: PSoC® Designer™ 5.1 SP2
Associated Application Notes: None

OBSOLETE

Abstract

AN2346 describes a low-cost, intelligent sensor network. The proposed network can be used for home and office automation, remote monitoring and control, securing and fire alarm systems, remote lighting control, and other applications that require low-cost, low-speed, bidirectional data communication. The advantage of PSoC® flexibility is demonstrated in the simplification of the project's hardware implementation.

Introduction

Continuing progress in microelectronic techniques and the incorporation of microcontrollers in all spheres of human activity push engineers to create systems that simplify products as well as save time and human resources. For example, the human presence is designed out in many production cycles, substituted by automatic machinery. This development results in increased productivity and improved system reliability. Similarly, it enables the realization of the "smart home."

Saturation of the measurement and control space with standalone remote microcontroller nodes results in the creation of large central dispatching panels to link these nodes to a system. Clearly, this node proliferation makes large systems complex, unwieldy and unreliable.

One of the first examples of distributed communication and centralized control for home networking is X-10, introduced in 1979. This protocol uses the home AC power lines for data transmission.

Today, the market for wireless networks (Wi-Fi, ZigBee, BlueTooth, IrDA, WirelessUSB, and others) is growing rapidly but there is still a demand for the development of wired networks. The primary weak point of wireless networks is the power supply; remote nodes must still be powered by local power or batteries, which must be charged from time to time. In addition to this, it can be asserted that the medium of data transmission, the radio channel, has considerably less noise resistance to interference than a wired channel. Battery powered wireless networks are less applicable in always-on systems where continuous monitoring and data transmission are required. The main advantages of wireless networks are assembly speed, node mobility, and economical installation (no cables). Wireless networks are well adapted to single time or mobile usage where the cost of wiring a network would be prohibitive. There is always the possibility of wire breakage in a wired network; installation time and cost are greater than for wireless networks, but if in service for a long time, the cost savings of not having to service batteries can be substantial.

In an effort to solve these problems, the wired sensor network was developed. Many such network implementations exist today, including Profibus, Modbus, CAN, LIN, X10, Linet, Ethernet, and LonWorks. Each has one or more of the following drawbacks:

- Relatively expensive, dedicated, modem/line arrangement chips must be used, boosting the overall network node cost.
- Network protocols can be complicated, and often require more expensive microcontrollers.
- Node power must be provided by separate wires, a requirement that increases installation and service costs.
- Network protocols cannot guarantee data delivery by a predicted time, and network performance is traffic dependent.
- Network may emit electromagnetic noise in a wide frequency range, which can cause electromagnetic compatibility problems or require costlier, shielded cables.

The proposed PSoC-based low-cost, intelligent network avoids the preceding drawbacks and has the following advantages:

- The Time-Triggered Protocol (TTP) ensures that data is delivered by the predicted time.
- Only two wires are used for bidirectional data communication and provision of power to the node.

- The quasi-harmonic, constant-frequency network signals are secure from electromagnetic compatibility and problems of certification.
- The nodes and host do not use proprietary or dedicated modem/line arrangement chips, so the intelligent, wired sensor network can be obtained at the price of a conventional wired sensor network or less because of PSoC advantages.
- Any PSoC device can be used as a network node controller, thus increasing design flexibility.

The developed network has the following characteristics:

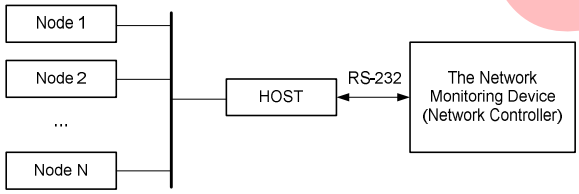
Table 1. Network Characteristics

Characteristics	Data
Carrier Frequency	20 kHz
Data Transmission Method	Each carrier period is used to transmit/receive one bit
Packet Length (PL)	PL = [3...30] bytes
Node Count (NC)	NC = [1..100] nodes
Power Supply	15 V AC, 4 A maximum

Operation Principles

The basis of the network is master-slave. There is one host and the necessary quantity of slave nodes, which execute given functions. The network structure is shown in Figure 1:

Figure 1. Sensor Network Structure



The network topology is common bus, which permits the use of inexpensive twisted pairs as the transmission medium. The various network components, shown in Figure 1, are as follows:

A single *host* supports protocol on the link layer and controls physical layer parameters. The host provides for data exchange between the nodes and the network controller.

The *nodes* of terminal devices execute specific functions. These nodes include sensor nodes for measuring the environment's physical parameters, and an actuating mechanism that provides for control of powerful machines, lighting systems, and so forth.

The *network controller* is connected to the host, and controls network resources. The network controller can be a PC or any other system with input/output capability.

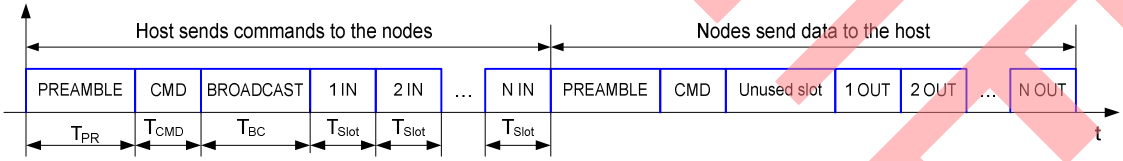
Several components of the sensor network theory play key roles in its function:

- Data Link Layer
- Physical Layer
- Network Packet Conception
- Network Period Calculation

Data Link Layer

The network protocol on the data link layer is shown in Figure 2.

Figure 2. Network Protocol Diagram



The network uses the collision-free TTP. Each node has its own time slot to receive or transmit data. The timeline protocol of a sample network is shown in Figure 2, which contains the following terms:

- The *PREAMBLE* synchronizes the nodes and host.
- The command identifier CMD determines the read and writes cycles.
- The *BROADCAST* messages are transmitted by the host to the nodes.
- *1 IN, 2 IN, …, N IN* is time slots (device addresses) during which the nodes listen to information from the host.
- The *Unused slot* 0 address during the host reception cycle corresponds to the BROADCAST slot during the transmission cycle. The host can use the unused slot

OBSOLETE

to inform the network controller (terminal program) about the network status.

- *1 OUT, 2 OUT, …, N OUT* are time slots (device addresses) during which the nodes transmit data to the host.

The protocol data exchange cycle has two stages:

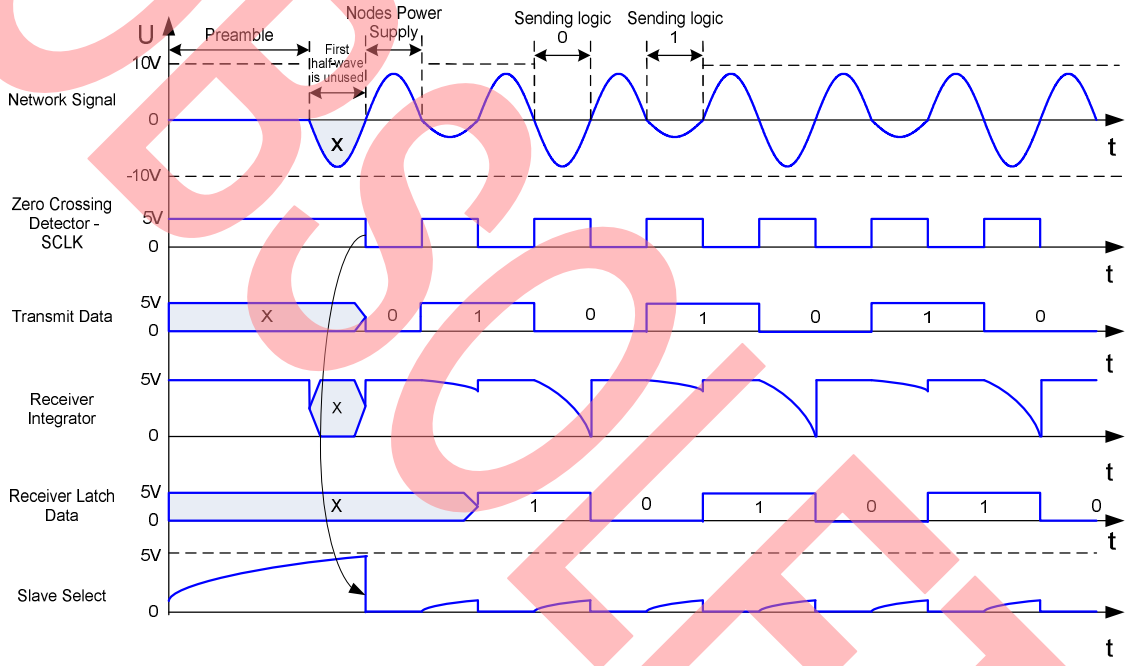
1. The host sends a broadcast message and data packets to the nodes.
2. The nodes reply to the host.

By means of the two stages, this collision-free TTP guarantees each node equal access to the physical channel for bidirectional data exchange.

Physical Layer

The network controller generates an amplitude-modulated constant carrier frequency. The upper half-wave provides power to the nodes. The lower half-wave is used for bidirectional data communication. The diagram in Figure 3 illustrates the timeline of the network during preamble, and some data bits. The scale of the U axis is not the same for all waveforms. For example, the Slave Select (SS) maximum voltage is 5 V, but its U length is not equal to the 5 V length of the zero-crossing detector waveform.

Figure 3. Selected Network Signals



Each carrier period is used for one bit of data transmission. The various amplitudes of the negative half-wave are used for information coding. Low amplitude is used to code the logic ‘1’ and high amplitude is used to code logic ‘0’. The carrier frequency is selected above the audio frequencies, but far below the radio frequency band. This selection method guarantees low electromagnetic radiation and eliminates the possibility of interference with other systems.

Data reception/transmission is implemented on the byte-oriented level. That is, the minimal information unit by which the node firmware can exchange information with the network is 1 byte (8 bits). The hardware transceiver SPI Slave (SPIS) User Module converts a byte to serial representation for its translation into the network and executes reverse actions during byte reception. The transceiver is implemented as hardware SPIS block. This approach only requires hardware implementation of the receiving/transmitting bridge and precludes the need for additional firmware support.

As shown in Figure 3, the preamble is presented as the absence of a signal in the line. During this time, the SS signal in SPIS is set to high. Consequently, this user module is reset and the node is initialized to work in the following cycle. The preamble time must not be too long, or the node supply capacitors will discharge to the CPU reset voltage level. After the preamble expires, the active protocol stage begins. The transceiver issues one data bit into the line at the beginning of the negative half-wave, and reads data into the RX latch at the upper half-wave. The first carrier half-wave is unused because before it begins, the SPIS is not initialized. The SPIS starts only after the SS signal becomes low. The SS falling edge occurs at the first positive half-wave, at the same moment that the first bit is issued into the line. At the next negative half-wave, the next bit is transmitted. At this moment the bit from the RX latch is transmitted to the SPIS shift register. But this bit is unknown at this time; that is, the first received bit is always indeterminate. That is why the bit that is received first from the network after the preamble is discarded. The SPIS generates an IRQ according to the

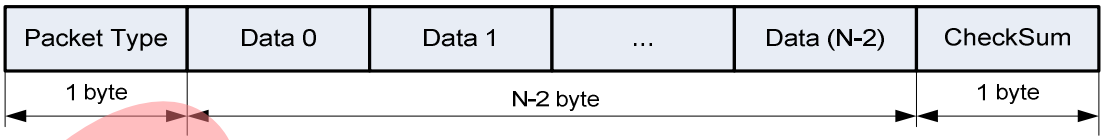
OBSOLETE

TxRegEmpty condition; that is, after the regular byte is received into the shift register. The interrupt handler reads the last value of the SPIS receive register and writes the next byte into the transmit register.

Network Packet Conception

All network transactions are carried out by data packets of fixed length. The packet structure is shown in Figure 4:

Figure 4. Network Packet Structure



The first byte determines the packet type. The last byte contains the packet checksum. All other bytes are data. The packet types can be service packets, for network configuration, or data packets, which are used on the application level. The broadcast messages do not have a specific packet type. These packets are transmitted to address 0, and therefore, are not needed in "type" assignment. They can be used for service protocol purposes and on the application layer. The packet length can be set by the user depending on the network application requirements. All necessary packet types can be set in the network configuration files.

Network Period Calculation

After the network is configured – that is, after node quantity, packet length, and carrier frequency are determined – the network period must be calculated. The equations for network period calculation are as follows:

$$T = 2 \cdot T_1 \qquad (1)$$

$$T_1 = \left[\left(PL \cdot (NC + 1) + 6 \right) * 8 + Preamble \right] \cdot \frac{1}{f_c} \qquad (2)$$

T is the full period of node polling. T_1 is the half-period of protocol (the read or write cycle in Figure 2). *PL* is the packet length expressed in bytes. *NC* is the address quantity (not counting the broadcast slot). *Preamble* is the length of the preamble expressed in carrier periods.

Six (*6*) is the quantity of additional service bytes needed for protocol functioning (this concerns work specific to the SPIS). Eight (*8*) is the number of bits in a byte. f_c is the carrier frequency.

The packet length and maximum number of nodes are tied by the following equation:

$$1536 = (NC + 1) \times PL \qquad (3)$$

The value 1536 determines the maximum buffer size on the host (CY8C29466). However, the quantity of nodes is determined not only by using Equation (3), but also by taking into account the maximum node power supply. Therefore, the maximum quantity is in the range of [50..100] nodes.

Thus, the period *T* determines the maximum latency time of packet delivery to its receiver.

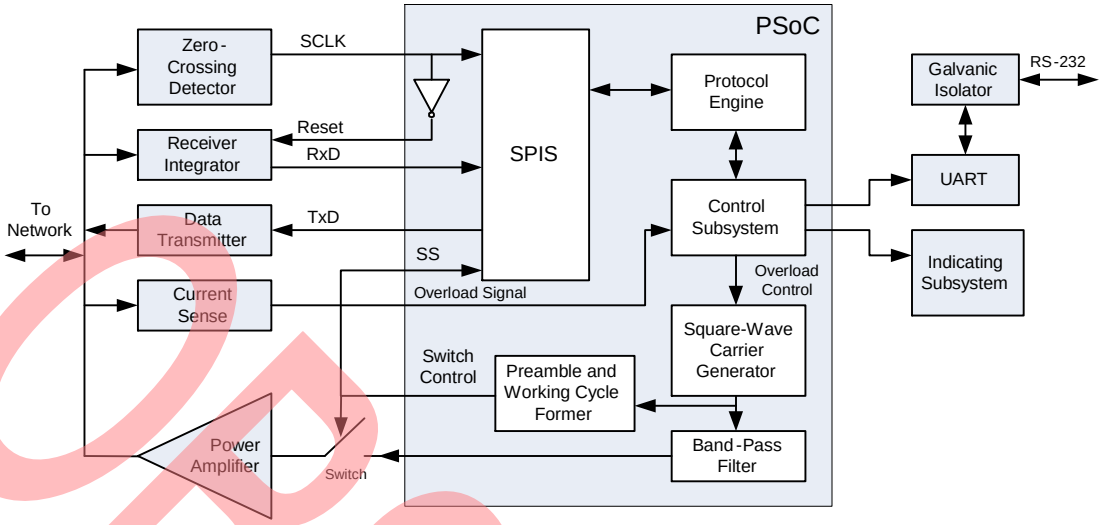
The Host

The host block diagram is shown in Figure 5.

The smart solution of network transceivers was derived from the hardware resources of the PSoC device and protocol characteristics. The heart of both host and node is the SPI Slave (SPIS) User Module. The SPIS at the hardware level provides a byte-oriented interface for data exchange between the firmware and the line. The separation of the bits and their transmission are placed entirely on the hardware. Processor time is saved, allowing the network carrier frequency to be above the normal audio range. This protects the network from electromagnetic compatibility and certification problems. Also, the CPU core clock rate is reduced and power consumption by the node decreases. In this implementation, the carrier frequency is set to 20 kHz.

OBSOLETE

Figure 5. Structural Schematic of the Network Host



Thus, the SPIS is a bridge between the data channel and the firmware of data processing (Protocol Engine). The Protocol Engine groups bytes into packets and places them in the buffers, analyzing packet types, and so on.

The Control Subsystem provides the initialization of all host subsystems, monitors the network to detect abnormal situations and their blocking, provides LED indication of current system status, and implements a data exchange protocol through RS-232 by means of the network controller. The Control Subsystem actually serves as a link that coordinates all work. The Galvanic Isolator provides safe electrical connections with other devices through a universal asynchronous receiver/transmitter (UART).

Note that the host functions do not include analysis of packet content. The host is a simple packet retransmitted.

The hardware portion of the host is implemented mostly inside the PSoC. Thus, for carrier generation, the output signal of the Square-Wave Carrier Generator feeds the input of the Band-Pass Filter, which removes harmonics from the transmitted signal. Next, to implement the data link layer, the preamble and working network cycle must be organized. The preamble is presented as the absence of carrier signal during some periods. This function is provided by the Preamble and Working Cycle Former, which is fed by the carrier signal in square-wave form. The Preamble and Working Cycle Former module calculates the preamble time during which the Switch is off; by this means, the carrier signal is kept from passing into the line. At all other times, the Switch is on. The Power Amplifier boosts the carrier signal to a suitable level. The Current Sense block protects the network from current overload and short circuits. Current Sense monitors moments of current overload in each carrier period, and Control Subsystem analyzes the quantity of overload situations during a set period of time. If the specified number of current peaks is exceeded, the Square-Wave Carrier Generator is shut down. The overload situation is indicated by a red LED, and the network controller is informed.

Three schematics of network connection bring the SPIS signals and network voltage levels into alignment: the Zero-Crossing Detector, Receiver Integrator, and Data Transmitter. The SS signal for the SPIS is generated by the Preamble and Working Cycle Former module. The integrator working cycle functions during the lower half-wave of the carrier. The succeeding upper half-wave resets the integrator.

In summary, the most distinctive features of the host in the PSoC-based, low-cost intelligent network include the following:

- Maximum transference of hardware functions into the PSoC, providing a byte-oriented interface with the firmware
- Protection against network current overload
- Carrier frequency out of the sound-wave range

These features are as beneficial during network use as they are during network development.

Host Hardware Implementation

The host electric circuit is described on two levels:

- PSoC Internal Configuration
- External Components

Advanced hardware resources have enabled implementation of many functional nodes inside the PSoC device. These functions include the following:

- Carrier Generator
- Preamble Formation Circuit
- Network Overload Protection

OBSOLETE

- UART and SPIS

Following are detailed descriptions of these nodes. Also see Figure 6, in which the PSoC internal user module placement and configuration are shown.

Carrier Generator

The Carrier Generator consists of the PWM8_1 and PWMDB8_1 digital blocks and one analog block, BPF2_1. The PWM8_1 sets the working frequency of the analog column with the band-pass filter, BPF2_1. PWM8_1 also is the source signal for PWMDB8_1, which controls the modulation of the BPF input signal. Thus, the carrier frequency is determined by the ratio of the output frequencies of PWM8_1 and PWMDB8_1.

Preamble Formation Circuit

This circuit is based in the PWM16_1, whose Period and PulseWidth parameters determine the durations of, respectively, the preamble and working network cycle. These parameters are presented in Equation (2) (the two items in square brackets). The input of PWM16_1 is driven by the DigBuf_1 of the signal from comparator bus 0 with BPF2_1 (see the Figure 6). The DigBuf is used because the comparator output is not directly available to the clock input of the PWM16_1. A block diagram of the *Carrier Generator* and *Preamble Formation Circuit* is shown in Figure 7 on page 8.

Network Overload Protection

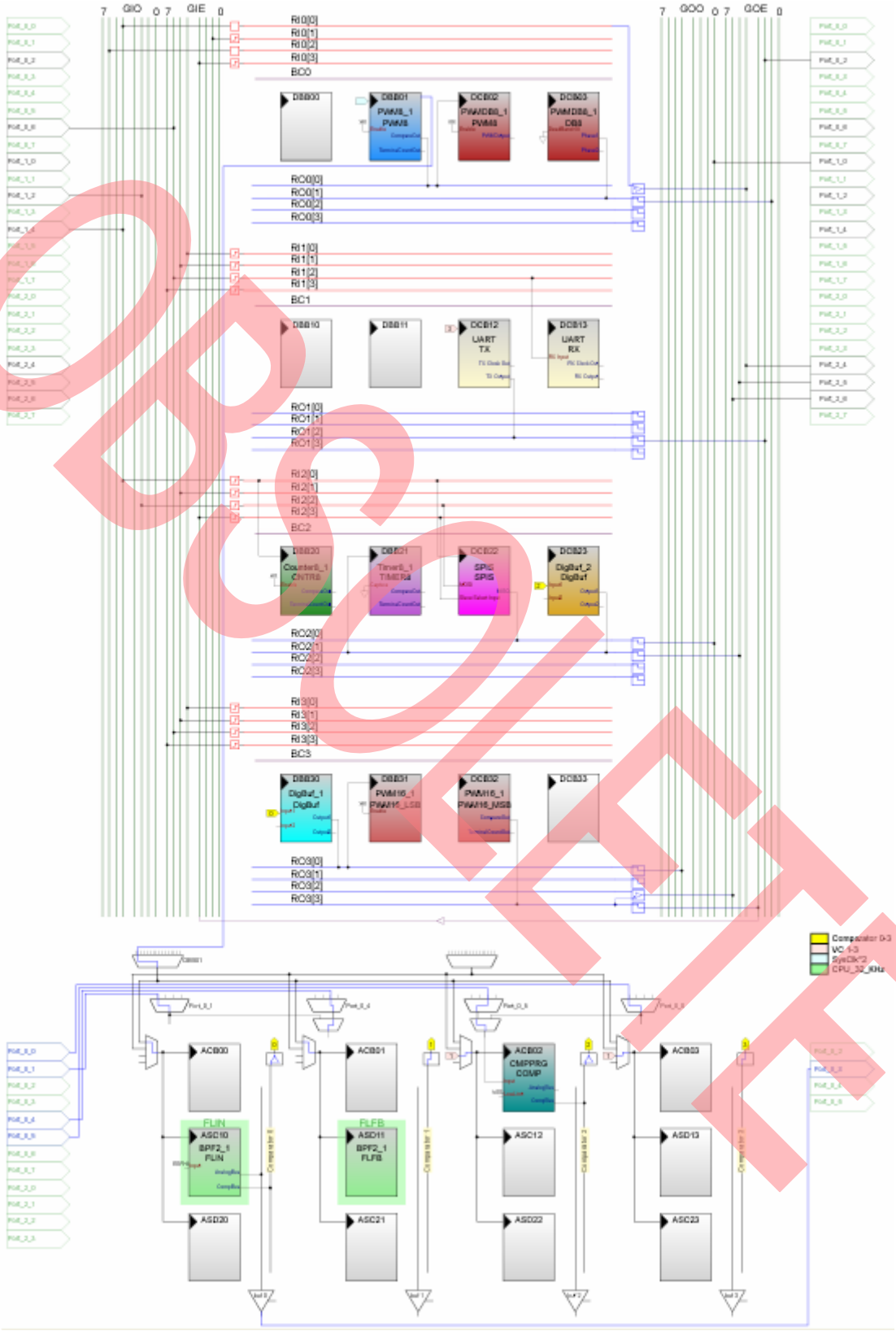
Network overload protection is implemented by using the Counter8_1, which generates interrupts 100 times per second (Hz). Its input signal is the carrier frequency after the *Zero-Crossing Detector*. Overload analysis is performed by the CMPPRG comparator. The input of the CMPPRG is fed by the voltage drop on the current sense resistor R7 (Figure 9). The comparator threshold is set by the LowLimit and RefValue parameters. This comparator is implemented according to the Schmitt trigger circuit, so that triggering occurs no more than one time per overload period. The number of overload periods is calculated by the Timer8_1. If the number of overload periods exceeds the admissible threshold, the system is stopped; otherwise, Timer8_1 is reinitialized. DigBuf_2 repeats the signal from the comparator bus and enables the red LED to indicate overload periods. The DigBuf is used because the comparator output is not directly available to the clock input of the Timer8_1. A block diagram of the overload protection circuit is shown in Figure 8 on page 8.

UART and SPIS

The UART function provides a connection to an external control device. It is set to a speed of 115.200 kbaud. The transceiver circuit is implemented in the SPIS. The SCLK, MOSI and MISO signals are routed outside the device, but the SS signal is formed internally by the PWM16_1.

OBSOLETE

Figure 6. PSoC Internal User Module Configuration



OBSOLETE

Figure 7. Block Diagram of Carrier Generator and Preamble and Working Cycle Former (PSoC Internals)

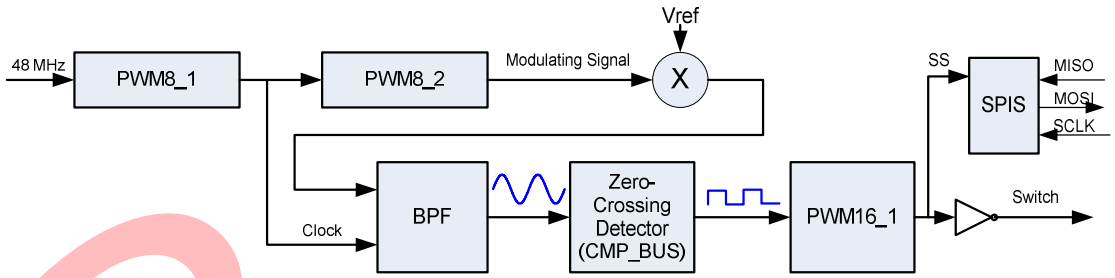


Figure 8. Block Diagram of Overload Protection (PSoC Internals)

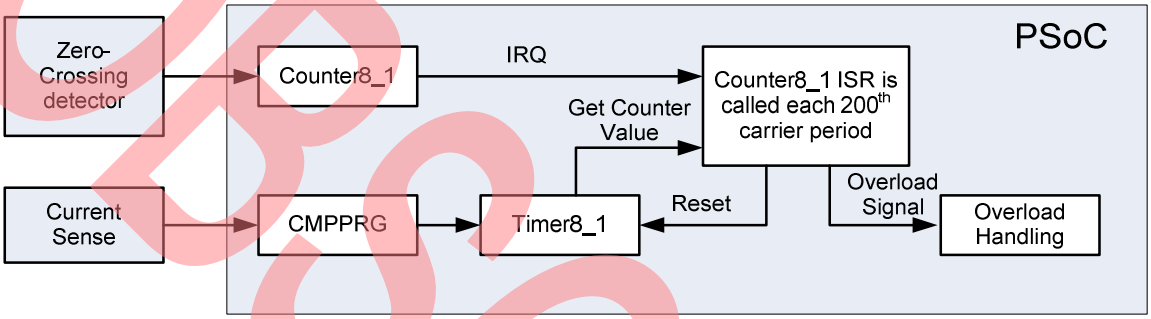
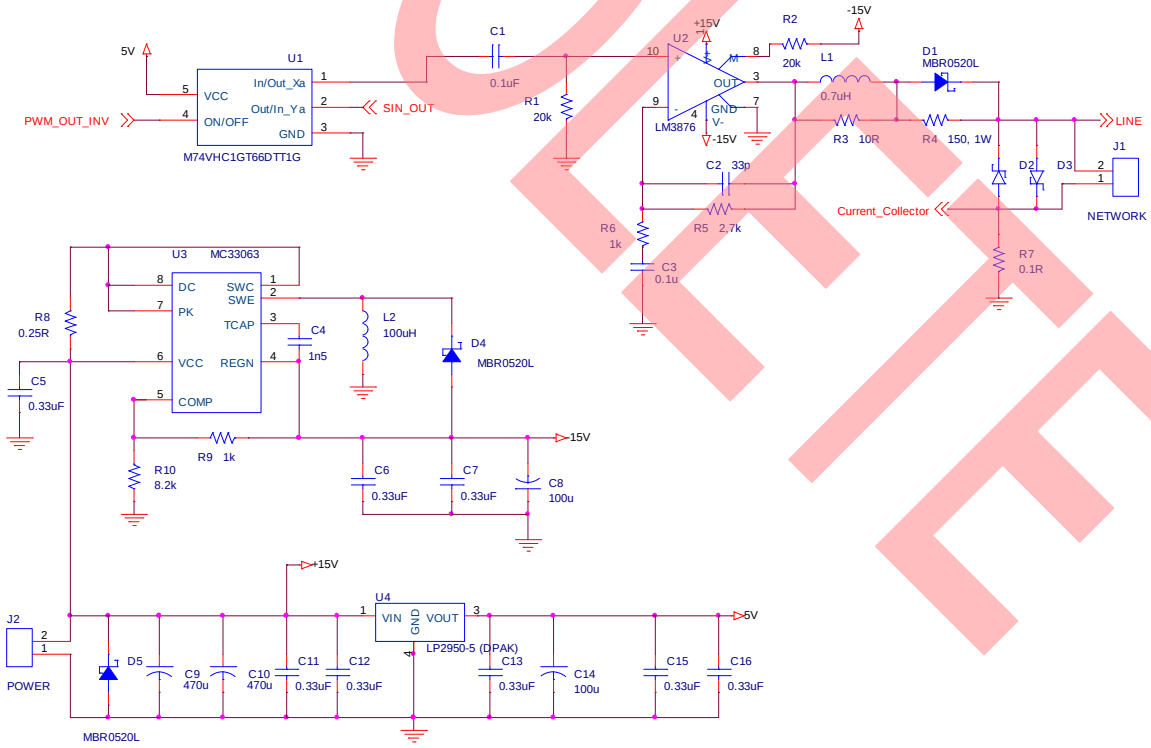
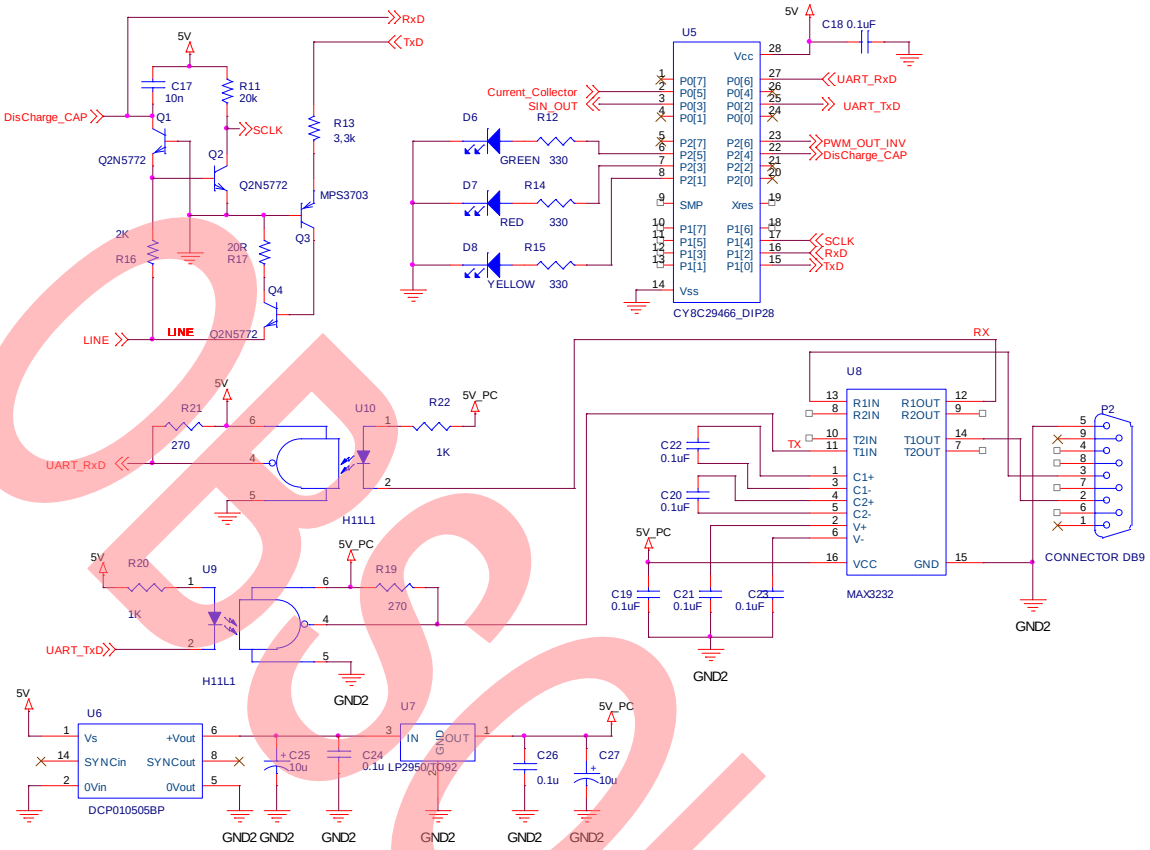


Figure 9. Host Analog Schematic



OBSOLETE

Figure 10. Host Digital Schematic



In Figure 9 on page 8, the analog section of the host circuitry is shown. In Figure 10, the digital section is shown.

The network is powered by a single +15 V power supply. The +5 and -15 voltages are produced by U4 and U3, respectively. The -15 V power consumption does not exceed several watts, so the low-power switching-mode inverter U3 is satisfactory.

The low-cost audio amplifier U2 scales the output signal to ±9 V_{pp}. For larger networks, higher output signal levels can be obtained by adjusting U2 (G=1+R5/R6) and by increasing the amplifier supply voltage. The matching circuit, consisting of L1, R3, D1, and R4, provides different impedance for upper and lower half-waves and ensures amplifier stability on heavy capacitance loads. The sense resistor R7 is used to measure network current consumption.

Preamble generation is implemented with analog switch U1.

The galvanic isolated UART consists of U6-U10. The power supply for the UART circuit on the PC side is provided by the DC/DC converter U6. The converter produces +5 V DC voltage, which is galvanically isolated from the host power supply. The voltage regulator U7 supplies output power voltage for U8 and U10.

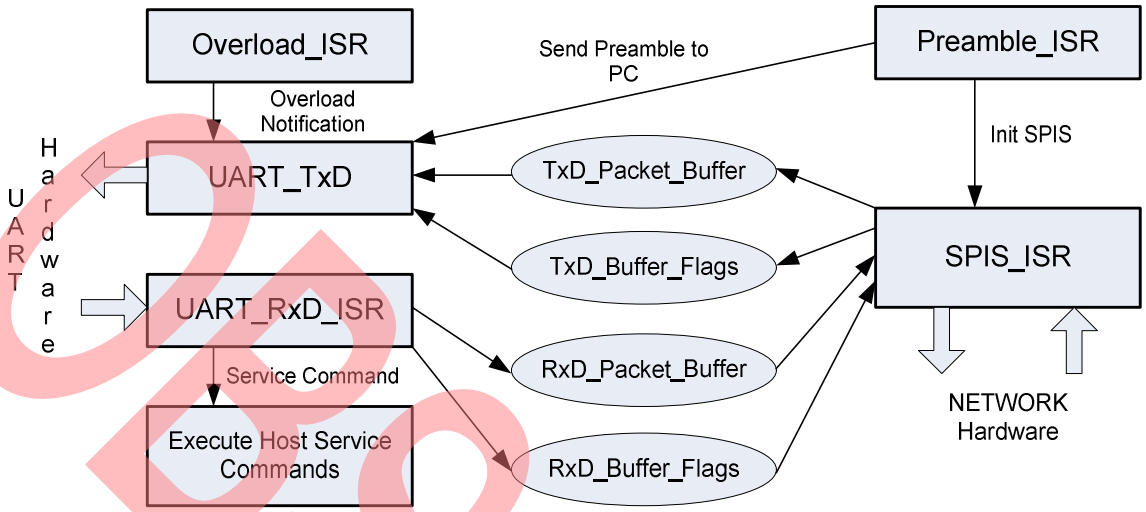
The zero-crossing circuit consists of the Q2 resistor and the R11 resistor. The zero-crossing detector Q2 forms the SCLK signal for the SPIS. The data transmitter is implemented on Q3-Q4, and the data receiver is formed by using Q1. The data receiver is based on the resettable integrator C17, which integrates the network voltage during the negative half-wave. By the falling edge of SCLK, the voltage is read on the MISO input of the SPIS. At the same moment, the integrator is reset by the Discharge_CAP signal, which is an inversion of SCLK. Discharge_CAP is connected to GPIO with an Open Drain High drive mode.

OBSOLETE

Host Firmware

The structure of interaction among the host firmware modules is shown in Figure 11:

Figure 11. Host Firmware



The host firmware consists of three basic parts, which are described ahead:

- UART Interface Servicing
- Protocol Function Implementation
- Network Overload Protection

UART Interface Servicing

Event handling is carried out primarily by an interrupt service routine (ISR) mechanism in the firmware. The data transceiver is implemented in SPIS_ISR, which is called each time after 1 byte is received from the line. SPIS_ISR also processes data: It forms packets, places them into buffers for the UART and performs reverse actions. In SPIS_ISR the packets are split into bytes and sent to the line.

Protocol Function Implementation

The Preamble_ISR initializes all protocol service variables, preparing the protocol for the next working cycle. Preamble_ISR also reinitializes SPIS, sets the preamble presence flag for the UART, and determines the direction of data transmission (from host to nodes or the reverse).

Network Overload Protection

Network overload protection is also implemented by ISRs. The Overload_ISR is called with a frequency of 100 Hz and analyzes the counter of carrier overload periods. If the counter value exceeds the admissible threshold, the overload flag for the UART is set and the network is stopped. Otherwise, the counter is reset.

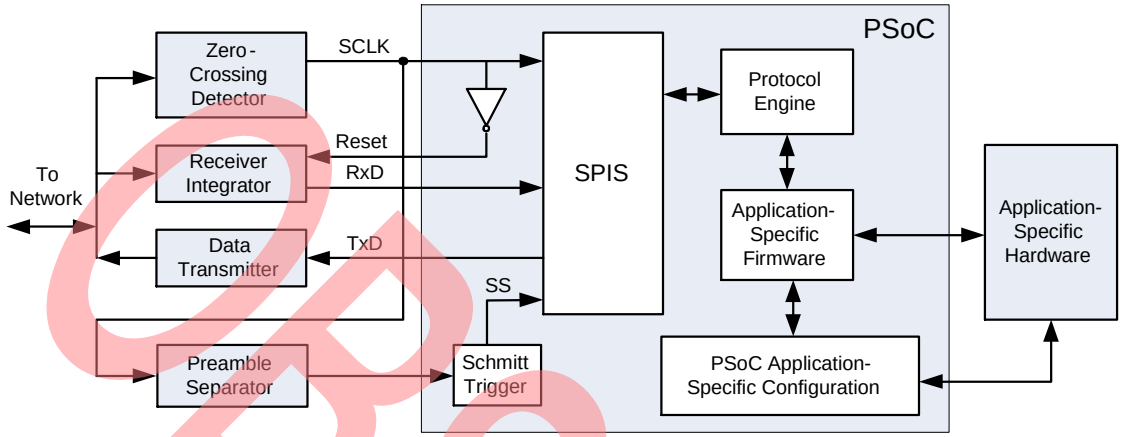
The buffers between SPIS_ISR and the UART temporarily save the data packets, which are translated on the other side of the host.

OBSOLETE

The Node

A block diagram of a model network node is shown in Figure 12:

Figure 12. Network Node Schematic



The implementation of the node transceiver section is very similar to implementation of the host transceiver section. The difference lies in the SS signal. In the case of the node, the SS signal is formed by an external integrator circuit. During carrier absence, the integrator is discharged. As a result, the IRQ signal is triggered, which reinitializes the SPIS and the Protocol Engine firmware, preparing it for the next working cycle.

A suite of user APIs is provided to facilitate data packet exchange between the network and node application. These APIs are accessible to the user program and include the following:

```
char P_send_packet(char* packet)
```

Sends a packet to the network

```
char P_receive_packet(char* packet)
```

Receives a packet from the network

```
char P_read_BC_msg(char* packet)
```

Receives a broadcast message from the line

Note that these functions are not blocking. That is, if a function is called, it copies a packet into the internal buffer and then returns to the point of the original call. This packet is transmitted, bit by bit, to the line by the SPIS_ISR. A function returns TRUE if a packet is received for processing, FALSE if the previous packet is currently being transmitted.

The `P_receive_packet` function returns TRUE if a packet is received from the network, FALSE otherwise – the program is not blocked while waiting for response from the line.

The Application-Specific Configuration and Application-Specific Hardware represent specific node functionality at the PSoC internal level and external component level, respectively. Designed and tested functions include a temperature sensor, lighting system control, and PIR detector. This group of functions could be expanded to include many kinds of applications.

Node Hardware Implementation

The node schematic described in this section is a common implementation for all node applications. The node controller can belong to any PSoC family depending on application requirements. In this implementation, the CY8C24xxxA chip series is used. Its internal configuration consists of one SPIS User Module. Thus, the protocol uses only one SPIS digital block as a transceiver. The remaining digital blocks and all analog blocks are resources intended for other application purposes.

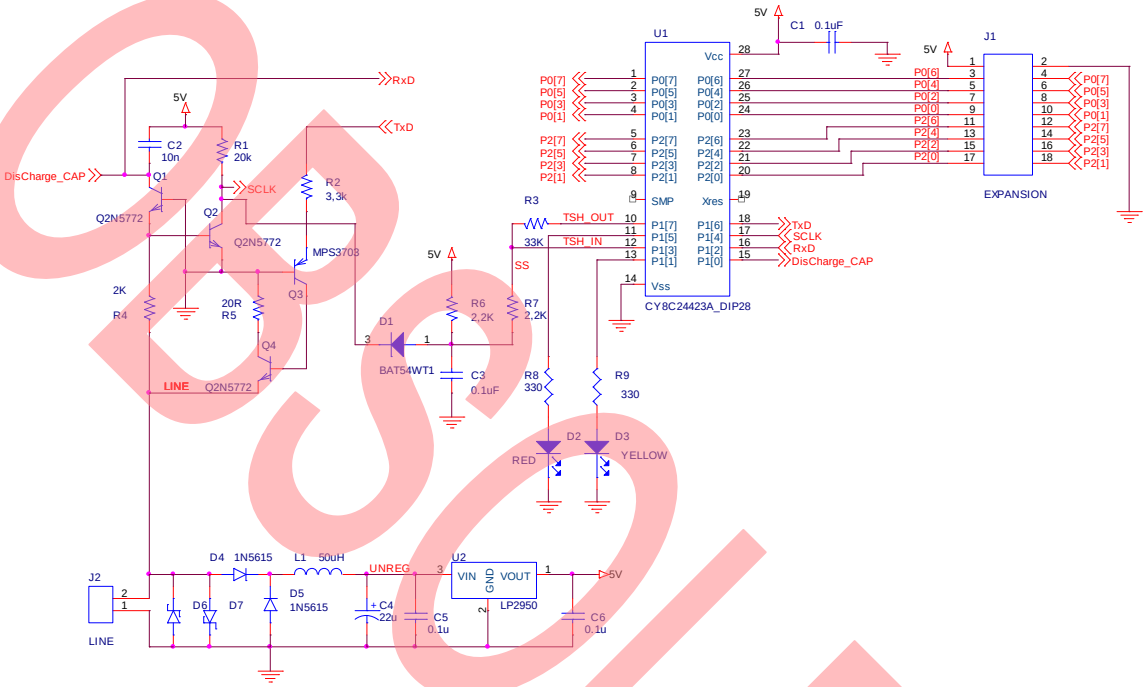
In Figure 13 on page 12, the electrical circuitry of the node is shown. The zero-crossing detector, receiver, and transmitter circuits are implemented in the same way that those of the host are implemented. The preamble integrator uses D1, R6, and C3. When the carrier signal is present, the voltage on C3 corresponds to logic ‘0’.

OBSOLETE

In the preamble period, C3 is charged to the voltage level of logic ‘1’, which causes GPIO interrupt triggering. The R3 and R7 resistors are the external components of the Schmitt trigger. The SS signal has a long rising edge that can cause multiple interrupt triggering during one preamble. The Schmitt trigger eliminates such possibilities. The D2 and D3 LEDs are useful for debugging.

All unused PSoC I/O pins are routed to the J1 slot for connection to the expansion board, on which the node application circuit is implemented.

Figure 13. Node Electrical Circuit

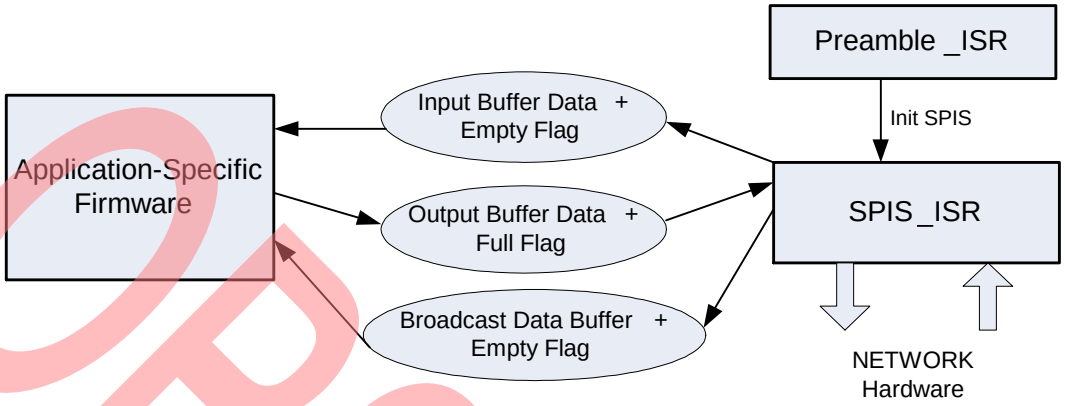


OBSOLETE

Node Firmware

The structure of interaction among the node firmware modules is shown in Figure 14:

Figure 14. Node Firmware Model



The node firmware consists of three basic parts:

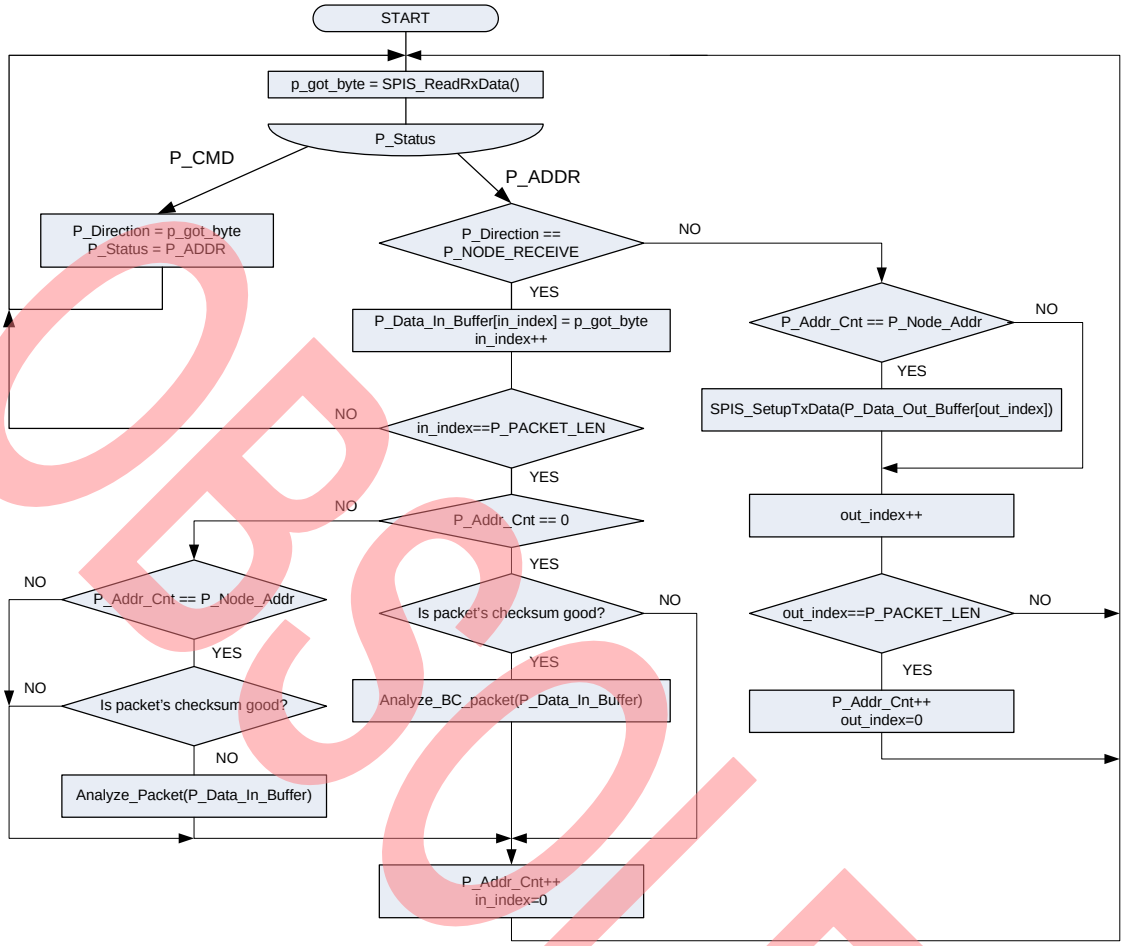
- Subroutines to access the physical layer (Preamble_ISR, SPIS_ISR)
- System buffers for data packets
- User firmware

User firmware utilizes API functions to exchange packets with the system buffers. The SPIS_ISR subroutine awaits its time slot (address), and then receives or sends a packet. After a preamble is detected, the Preamble_ISR is called and initializes the protocol for work in the next network cycle.

An SPIS_ISR flowchart is shown in Figure 15 on page 14. The subroutine is implemented as a finite state machine.

OBSOLETE

Figure 15. SPIS Transceiver ISR Firmware Flowchart



The P_Status variable determines the type of data received. There are two types. The P_CMD byte identifies the direction of transmission (see Figure 2 on page 2), and the P_ADDR data bytes are addressed to either the nodes or the host.

The P_Addr_Cnt variable saves the current address of a translated packet in the line. P_Node_Addr is the node address; it must be constant and assigned during the chip programming stage. The p_got_byte variable is the last byte received from the network.

The variables of this ISR are initialized after the preamble is detected in the Preamble_ISR.

Summary

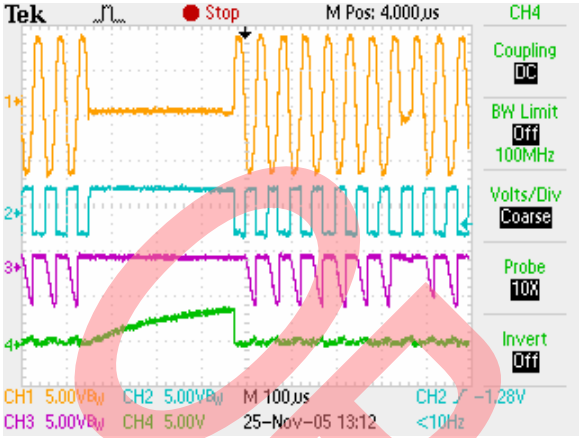
The multi-purpose, low-cost, intelligent sensor network is presented. The network implementation can serve as the basis for a range of commercial products. The Open Systems Interconnection Reference Model (OSI Model) is presented as a physical layer, data link layer, and application layer. This is the requisite model for network use and future development.

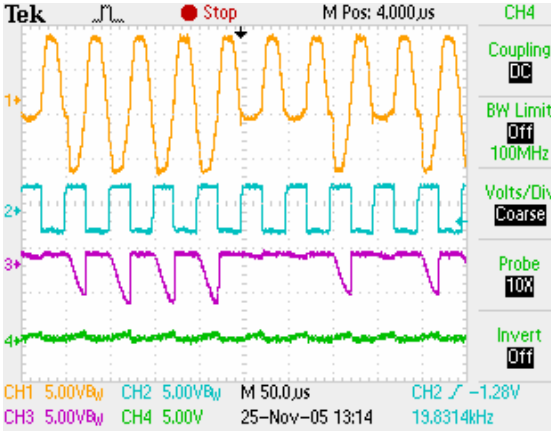
To design a system that has the necessary functionality, the network controller must be developed. The network controller can be a PC with a corresponding program, or some other device that has a UART interface with implementation of system logic.

Plug and Play (PnP) and sleep mode features can be added in commercial implementations. The current design assumes the use of hardware-adjusted address for the node. Use of the low-cost sensor network is limited only by the user's imagination or specific engineering requirements.

OBSOLETE

Appendix: Network Oscillograms





Notes

1. Signal on Communications Line (Carrier + Information Bits) (Orange)
2. SCLK (Blue)
3. Receiver Integrator (Purple)
4. SS Integrator (in the Node) (Green)

OBSOLETE

About the Author

Name: Andrew Smetana
Title: Electrical Engineer
Background: Andrew received a Masters of Science degree from National University “Lvivska Polytechnika“ (Ukraine) in 2004. His interests include various aspects of embedded systems development.
Contact: andi@cypress.com

Document History

Document Title: Intelligent Sensor Network – Physical and Data Layers – AN2346

Document Number: 001-25599

Revision	ECN	Orig. of Change	Description of Change
**	1444663	GJV	New Application Note
*A	3041663	GJV	Sunset Review
*B	3357824	ANBI_UKR	Modified title. Project support the latest version PSoC designer
*C	4481158	BOO	Obsolete

OBSOLETE

In March of 2007, Cypress recataloged all of its Application Notes using a new documentation number and revision code. This new documentation number and revision code (001-xxxxx, beginning with rev. **), located in the footer of the document, will be used in all subsequent revisions.

PSoC is a registered trademark of Cypress Semiconductor Corp. "Programmable System-on-Chip," PSoC Designer, is trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408-943-2600
Fax: 408-943-4730
http://www.cypress.com/

© Cypress Semiconductor Corporation, 2007-2011. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

This Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.