Sensors and Materials, Vol. 35, No. 11 (2023) 3623–3636
MYU Tokyo

S & M 3440

Developing a Monitoring System for Wire Bonders Using Semiconductor Equipment and Materials International (SEMI) Equipment Communications Standard Sensors

Ying-Chieh Liu[1] and Feng-Chia Li[2*]

[1]Department of Information Management, National Chin-Yi University of Technology
No. 57, Sec. 2, Zhongshan Rd., Taiping Dist., Taichung 411030, Taiwan (R.O.C.)
[2]Department of Distribution Management, National Chin-Yi University of Technology
No. 57, Sec. 2, Zhongshan Rd., Taiping Dist., Taichung 411030, Taiwan (R.O.C.)

(Received August 31, 2023; accepted October 30, 2023)

Keywords: semiconductor, assembly, wire bonding, SECS, sensor

Wire bonding is one of the important assembling processes in the semiconductor manufacturing industry. It is responsible for connecting a good die to a lead frame using automatic equipment called wire bonders. Among all the assembling processes, wire bonding is one of the most time-consuming processes. Therefore, it is crucial to monitor the states of wire bonders in order to efficiently repair and maintain them. This becomes especially important when there are numerous wire bonders in the production line, as it helps improve productivity. In this study, we developed an information system that communicates with wire bonders using Semiconductor Equipment and Materials International (SEMI) Equipment Communications Standard (SECS) sensors. The SECS sensors, which combine the SECS protocol and the functions of the equipment automation program (EAP), take charge of receiving and responding messages from wire bonders and hosts, and acts as a bridge to help collect information on wire bonders and transmit it to our system. The developed system can show the states and error messages of wire bonders, wires, and bond heads. It also provides historical records and availability information of wire bonders. This assists managers in making informed decisions to reduce the time and effort required for maintaining wire bonders. Overall, our system effectively enhances the monitoring and maintenance processes of wire bonders, leading to improved productivity in assembling production.

1. Introduction

There are four main steps in semiconductor production: wafer design, wafer making, assembling, and testing. The purposes of assembling are to saw the wafer into die and assemble die as an integrated circuit (IC). The key processes of assembling include back grinding, wafer sawing, die attachment, wire bonding, molding, and marking.(1) Among these processes, wire bonding is the most time-consuming one.(1) It involves connecting a good die to a lead frame

*Corresponding author: e-mail: lear999@ncut.edu.tw
https://doi.org/10.18494/SAM4643

using wire bonders. This process establishes an electric current pathway by connecting the leads on the lead frame to the die. Bonding wire materials can include aluminum, gold, copper, and silver. Previous studies have explored various technologies to improve the wire bonding process, such as new bonding techniques,(2) novel wire materials,(3) AI inspection systems for visual defect detection,(4) and analysis systems for quantitative fault analysis in the die attachment process.(1)

However, apart from enhancing the machinery and materials, it is also crucial to reduce the time and effort required for repairing and maintaining wire bonders in order to achieve an efficient bonding process and ensure high availability, particularly when dealing with numerous wire bonders. Limited research has been conducted on monitoring wire bonders to enhance the efficiency of engineer dispatch and minimize repair and maintenance time. Thus, we developed an information system that monitors the states of wire bonders by communicating with Semiconductor Equipment and Materials International (SEMI) Equipment Communications Standard (SECS) sensors and established a situation room for managers to address breakdowns and failures of wire bonders. In this study, the SECS sensors were modeled as sensors combining the equipment automation program (EAP)(5) and SECS, facilitating communication between semiconductor equipment and computer information systems. SECS is a communication protocol designed for semiconductor equipment, and it requires EAP to realize the transmission and execution of physical packets. EAP is used to control equipment materials and operations, and when integrated with SECS, it achieves the goal of equipment automation. The SECS sensors serve as intermediaries for communication between semiconductor equipment and the information system. Our system connects the SECS sensor built in wire bonders to obtain and show their states, allowing managers to monitor all wire bonder states and implement efficient methods to enhance wire bonder availability. Overall, this study presents an information system that effectively monitors wire bonder states, facilitates efficient maintenance, and improves wire bonder availability.

2. Literature Review

2.1 SECS

SECS is an equipment-to-host communication protocol used in the semiconductor industry and managed by the association of SEMI. It encompasses standards E4, E5, E30, and E37, which specify communication aspects ranging from physical layer communication (RS232) to multisession communication (TCP/IP).(2) Furthermore, SECS offers both message-based and scenario-based communication, facilitating clear and logical exchanges between the equipment and the host.

Figure 1 illustrates the structure of the SECS layer protocol, with each layer described below.

(1) Physical link and data transfer protocol:
SECS offers two methods, SECS-I and High-Speed SECS Message Services (HSMS), for linking equipment and the host. Both methods define electrical specifications, transmission speed, handshake codes, message length, message header, checksum, and waiting time. SECS-I is used for low transmission rates and utilizes RS-232, whereas HSMS is used for high transmission rates and employs TCP/IP.

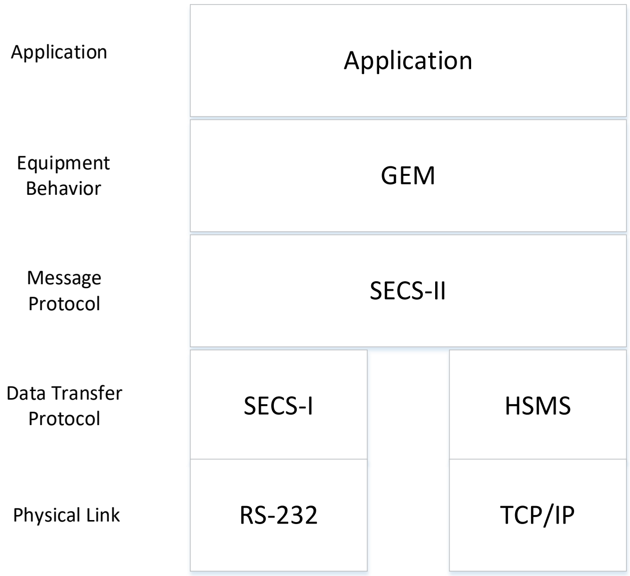

Fig. 1. SECS layer protocol.

(2) SECS-II:
SECS-II provides the definition of messages and related data items exchanged between the equipment and the host. All messages are categorized on the basis of their attributes, referred to as streams. For example, there are streams such as equipment status (S1), data collection (S6), and recipe management (S15). Each stream consists of numerous assigned messages, known as functions. Every message is composed of individual items or lists of items. Table 1 shows the list of SECS-II streams.

(3) Generic Equipment Model (GEM):
GEM defines which SECS-II messages should be used in specific situations and specifies the resulting activity in order to describe the scenarios of equipment behaviors. It outlines the appropriate usage of SECS-II messages, identifies the situations where they are applicable, and determines the corresponding activities or actions that should be undertaken by the equipment.

2.2 HSMS

The format of an HSMS packet, as depicted in Fig. 2, consists of several bytes. The first 0 and 1 bytes of the header represent the equipment number, with all message headers using an equipment number of 0xFFFF. The second byte indicates the Stream ID of the message and whether the message requires a response. The third byte represents the function ID of the message, which, along with the second byte, determines the message type. The fourth byte represents PType, with a value of zero indicating adherence to SECS-II rules. The fifth byte represents SType, with a value of zero indicating that the message contains transmitted information. SType values ranging from 1 to 9 correspond to control messages. Bytes 6–9 are

Table 1
List of SECS-II streams.

No.	Stream	No.	Stream
1	Equipment status	9	System error
2	Equipment control and diagnostics	10	Terminal services
3	Material status	11	Host file services
4	Material control	12	Wafer mapping
5	Exception reporting (Alarms)	13	Unformatted data set transfers
6	Data collection	14	Object services
7	Process program management (Recipes)	15	Recipe management
8	Control program transfer		

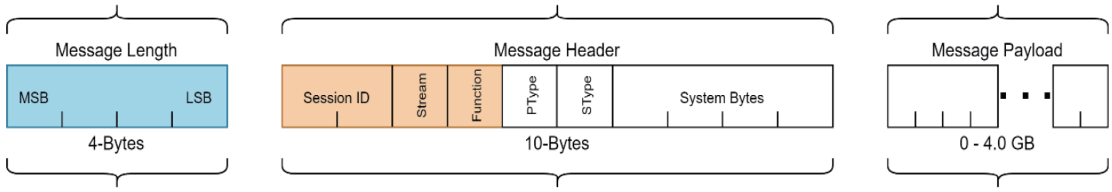

Fig. 2. (Color online) HSMS packet format.(6)

system bytes used to differentiate messages. Each message has a unique system byte to distinguish it from others. The system byte of the return message must match that of the main message. In a control message, the system byte of the response message must also match that of the request message. The SType byte is typically utilized to differentiate control message types, whereas the PType byte is consistently set to zero. However, for return messages in control messages, the third byte represents the state value of the control message, and in reject messages, it signifies the reason code. The Linktest control message has an equipment number of 0xFFFF.

HSMS serves as an alternative to SECS-I in cases where rapid communication is required or when a simple point-to-point topology is insufficient. It defines a communication interface suitable for exchanging messages between the equipment and the host within a TCP/IP environment. The HSMS standard (SEMI E37) comprises two substandards: SEMI E37.1 and SEMI E37.2. The former employs a single session (HSMS-SS) for typical equipment using SECS-II with or without GEM, whereas the latter is specified for the MESC/CTMC protocol on vacuum cluster equipment.

2.3 SECS-II

SECS-II provides a set of data structure rules and message structure directions that enable communication and data exchange between the equipment and the host. The data structure rules are as follows.
(1) All data in a message are contained in items.
(2) An item is one or more data values of the same data format.
(3) There are 15 data formats of items recognized by SECS-II.

(4) A SECS message contains at least one item.
(5) If a message contains more than one item, the items are grouped in a list.
(6) A list may contain items and other lists.

Figure 3 illustrates the format of a SECS-II packet. The first message (S6F9) is used to transmit data from the equipment to the host. The second message (S6F10) is a response to the S6F9 message sent by the host

S6F9 Equipment to Host PFCD = 1

Byte	Hex	Value	Description
1	80		R-bit = 1 Host ← Equip
2	00	Device ID = 0	
3	86	Stream 6	W-bit = 1 Reply Expected
4	09	Function 9	
5	80		E-bit = 1 Last Block
6	01	Block # = 1	
7	XX		System Bytes
8	XX		
9	XX		
10	XX		
11	01	List	
12	04	4 items	
13	21		Item 1 format (binary)
14	01		Item 1 length
15	01	PFCD = 1	
16	69	2-byte signed	Item 2 format
17	02		Item 2 length
18	00		
19	06	DATAID = 6	Load C Pumpdown Started
20	21		Item 3 format
21	01		Item 3 length
22	01	CEID = 1	
23	01	List	Item 4
24	00		Zero items

S6F10 Host to Equipment

Byte	Hex	Value	Description
1	00		R-bit =0 Host → Equip
2	00	Device ID = 0	
3	06	Stream 6	W-bit = 0 No Reply
4	0A	Function 10	
5	80		E-bit = 1 Last Block
6	01	Block # = 1	
7	XX		System Bytes
8	XX		
9	XX		
10	XX		
11	21	Binary	Item 1 format
12	01	1 byte	Item 1 Length
13	20	Ack6 = 0	Acknowledge code (OK)

Fig. 3. Format of SECS-II packet.

3. System Design

3.1 SECS sensor and data transmission

Wire bonders must be equipped with the SECS sensors that support the SECS-II and TCP/IP communication modules. The sensor and data transmission structure are illustrated in Fig. 4.

The introduction of the SECS sensors and data transmission is as follows.

(1) Controller: The controller serves as the center unit that controls the pressure, temperature, and mechanism of wire bonders. It also displays information on the user interface and receives commands from operators.

(2) User interface: It is used to display the status of wire bonders, receive commands from operators, and communicate with the controller.

(3) SECS-II module: The SECS-II module is responsible for parsing messages from the host and communicating with the controller to obtain the relevant information. It encodes the information into the SECS-II message format and sends it back to the host based on the purpose of the messages.

(4) Network communication module: The network communication module encapsulates the SECS-II message in the HSMS format and transmits it over the TCP/IP network. It works in conjunction with the SECS-II module to form the SECS sensors, facilitating communication with the system developed in this study.

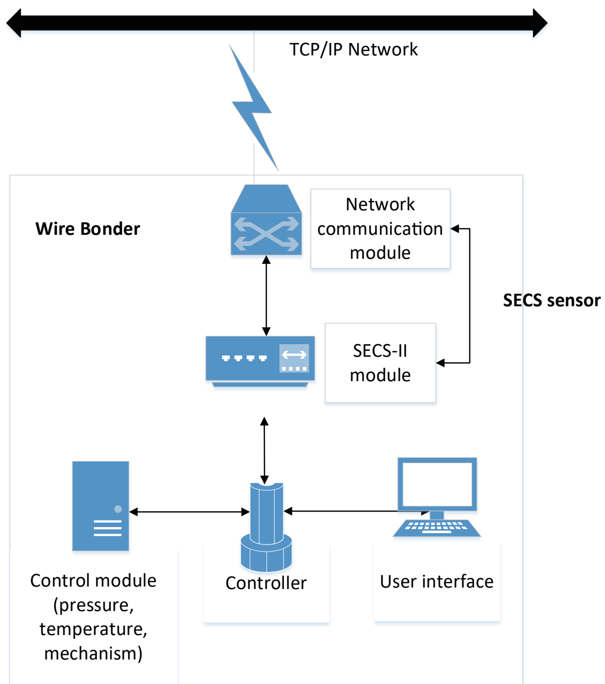

Fig. 4. (Color online) Sensor and data transmission structure of wire bonder.

3.2 Scenario of communicating between host and wire bonder

The purpose of this system is to collect and display the status of wire bonders, assisting managers in improving the availability of wire bonders. Thus, the most important scenarios are communication and event report.

3.2.1 Communication state model

Figure 5 illustrates the communication state model, whereas Table 2 shows communication state transitions. The default of the communication state can be either enabled or disabled. If the state is enabled, the communication between the equipment and the host can be established using S1F13 and S1F14 (see Table 3). Otherwise, S1F1 and S1F2 can be used to determine the communication state (see Table 4).

In the enabled/not communicating state, wire bonders only accept S1F13 and stream 9 messages. When wire bonders receive S1F13, they respond with S1F14. If wire bonders are ready to establish communication, then Commack is set to 0. Otherwise, Commack is set to 1. When S1F14 is sent successfully, wire bonders change the states from enabled/not communicating to enabled/communicating, and the control state is set to the default state. If wire bonders receive S1F13 from the host and the states are already set to enabled/communicating, then wire bonders respond with S1F14. If S1F13 and S1F14 are not being used to establish communication, the states are always communicating. A status variable (CommEnabled) is used to determine whether or not wire bonders have disabled communication. When wire bonders receive S1F1 and communication is enabled, they reply with S1F2. If communication is disabled, wire bonders do not respond to any messages received from the host.

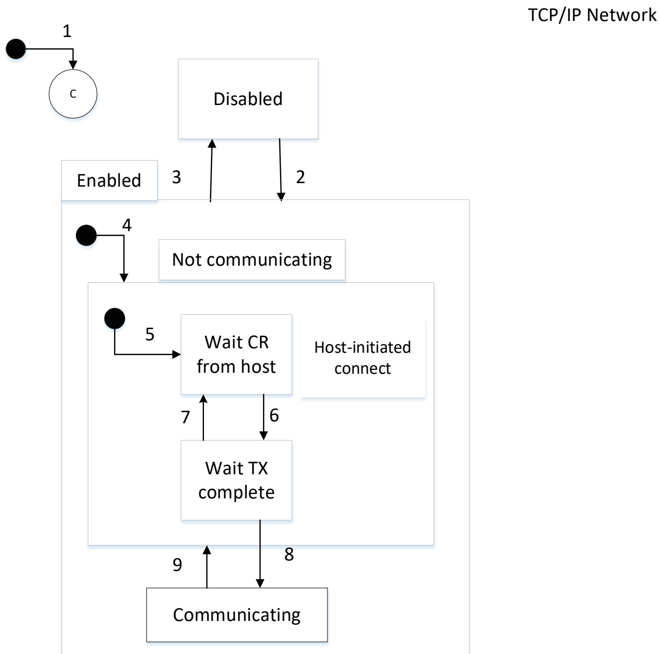

Fig. 5. Communication state model.

Table 2
Communication state transitions.

No.	Current state	Trigger event	New state	Action	Comment
1	Entry to communication	System initialization	System default	None	Default may be disabled or enabled
2	Disabled	Operator switches from disabled to enabled	Enabled	None	SECS-II communication is enabled
3	Enabled	Operator switches from disabled to enabled	Disabled	None	SECS-II communication is prohibited
4	Entry to enabled	Any entry to enabled state	Not communicating	None	May enter from system initialization to enabled or through operator switch to enabled
5	Entry to host-initiated connect	Any entry to not communicating	Wait CR from host	None	Wait for S1F13 from host
6	Wait CR from host	Receive S1F13	Wait TX complete	Send S1F14 with Commack = 0	Host seeks to establish communications
7	Wait TX complete	S1F14 transmission failed	Wait CR from host	None	Wait for S1F13 from host
8	Wait TX complete	S1F14 transmission completed successfully	Communicating	None	Communication is established
9	Communicating	Communication failure	Not communicating	Dequeue all messages queued to send	Dequeued messages may be placed in spool buffer as appropriate

Table 3
Scenario: Host attempts to establish communication.

Comment	Host	Equipment	Comment
			Communication state is enabled
Establish communication request	S1F13 →		
		← S1F14	Reply COMMACK = 0 and communication state = communicating

Table 4
Scenario: Host-initiated are you there request.

Comment	Host	Equipment	Comment
Are you there request	S1F1 →		
		← S1F2	On-line data response

3.2.2 **Process state model**

Figure 6 shows the process state model for wire bonders. There are four states in the model:

(1) Auto run: RUN button is pressed and wire bonders are running.

(2) Pause: Wire bonders are in the auto run state and the operators manually press the STOP button.

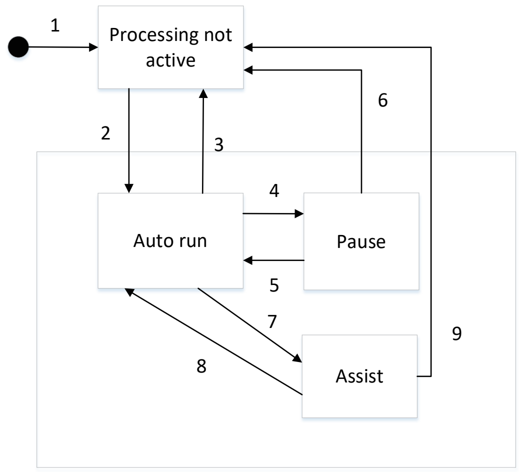

Fig. 6. Process state model.

(3) Assist: Wire bonders are in the auto run state and an assist condition occurs.
(4) Processing not active: All other machine states.

Table 5 shows process state transitions. When wire bonders power up, they enter the state of processing not active from system initialization (No. 1). After the operators press the RUN button, wire bonders go to the state of auto run from processing not active (No. 2). When the job is done or a fatal error occurs, the state is changed from auto run to processing not active (No. 3). The switch between the states of auto run and pause is due to the operators pressing the STOP and RUN buttons (Nos. 4 and 5). If the operators select ABORT AUTO from sequence stop or standby state, wire bonders enter processing not active from pause (No. 6). When wire bonders detect an assist condition, they change to the state of assist (No. 7), and the state of assist changes to auto run if the operators resolve the issues (No. 8). However, if the operators select ABORT AUTO or standby mode, wire bonders change to the state of processing not active from assist (No. 9).

There are two methods used to collect the states of wire bonders: reported by wire bonders (see Table 6) and requested by the host (see Table 7). The former occurs when wire bonders change the states, and they send S6F5 to the host. The host sends S6F6 to grant wire bonders to report the states. Then, wire bonders send S6F11 to report their states. At the end, the host sends S6F12 to acknowledge the event report. The latter occurs when the host requests wire bonders to send their states. The host sends S6F15 to wire bonders and wire bonders reply S6F16 back to the host to report their states.

3.3 Classifying the events

We categorized events from wire bonders into four parts: green, yellow, red, and unknown. Since different brands and types of wire bonder may have different events, we classified the known events into green, yellow, and red categories. When unknown events are received, they are assigned to the unknown classification and await engineers to classify and assign them to the

Table 5
Process state transitions.

No.	Current state	Trigger event	New state	Action	Comment
1	System initialization	System power-up	Processing not active	None	
2	Processing not active	Operators press run button and wire bonders enter auto mode.	Auto run	None	
3	Auto run	Job completed successfully or a fatal error occurs	Processing not active	None	
4	Auto run	Operators press stop button and wire bonders enter sequence stop	Pause	None	
5	Pause	Operator resumes auto bonding	Auto run	None	
6	Pause	a. Operators select abort auto from sequence stop b. Operators select standby mode	Processing not active	None	
7	Auto run	Auto detects an assist condition	Assist	None	
8	Assist	Operators complete assist	Auto run	None	
9	Assist	a. Operators select abort auto from assist b. Operators select standby mode	Processing not active	None	

Table 6
Scenario: Event data collection from wire bonder.

Comment	Host	Wire bonder	Comment
		← S6F5	Wire bonders change the states
grant wire bonders to send the states	S6F6 →		
		← S6F11	Send event report (states)
Acknowledge the event report (states)	S6F12 →		

Table 7
Scenario: Host requests collection event report.

Comment	Host	Wire bonder	Comment
Host requests collection event report	S6F15 →		
Send multiblock grant		← S6F16	Wire bonder sends a collection event report

correct category. Owing to the large number of events, Table 8 shows a small portion of the events for the green, yellow, and red categories. Green messages indicate that wire bonders are running smoothly. Yellow messages indicate that wire bonders are in an alarm state but no immediate intervention is needed. Red messages indicate that engineers or operators need to intervene or repair the wire bonders.

3.4 System structure diagram

Our system structure diagram is shown in Fig. 7. The system structure is explained below.

(1) SECS-II module:
This module plays a crucial role in our system. It is responsible for receiving messages from wire bonders, responding to messages from the host, and periodically sending state requests

Table 8
Classifications of green, yellow, and red.

Classification	Event id	Name	Description
Green	8001	Control state changed to online	Wire bonders change to the state of online
	8004	Process state change	Wire bonders change state
	8022	Operator data entry	When operators enter data
Yellow	6077	PDT-wirefeed-error	PDT-wirefeed-error
	9140	Process-error	Process-error-set
	9770	PUL_carrier_not_present	Empty input magazine slot
Red	6018	Ind_carrier_jam	Lead frame jammed during ejection
	6059	Auto_bondhead_safety_violatioin	Bond head could not be positioned without exceeding table limits
	10710	MH_touch_active_invalid_abort	Input mag handler touch sensor blocked

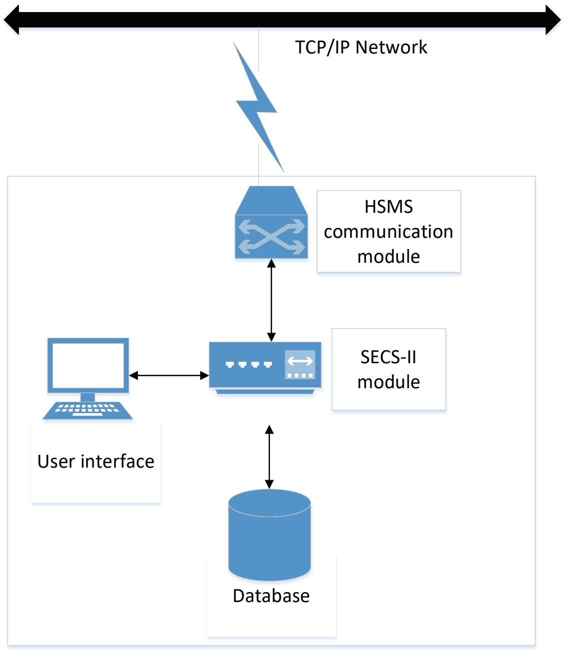

Fig. 7. (Color online) System structure diagram.

to wire bonders using a polling policy. The states of wire bonders are displayed on the user interface.

(2) HSMS module:
This module handles packet communication. It receives SECS-II messages, converts them into HSMS format packets, and sends them to the TCP/IP network. It also receives messages from wire bonders and forwards them to the SECS-II module.

(3) User interface:
The user interface primarily receives messages from the SECS-II module and displays the states of wire bonders. When users need to check historical data, it requests the SECS-II module to retrieve the data and display it in a user-defined format.

(4) Database:
A database is used to store the history of states of wire bonders.

4. System Implementation

We developed the system using C# and referenced the codes provided by Hume Integration Software.(7) Since the wire and bond head are the components that experience frequent errors in wire bonders, thus in addition to monitoring the states of wire bonders, we also collected the states of the wire and bond head. This helps to reduce the time required for repairs and maintenance.

Figure 8 illustrates the scenario of 'host-initiated are you there' communication. Our system sends the S1F1 message to wire bonders, and wire bonders respond with the S1F2 message to establish communication. Figure 9 depicts the scenario of 'host requests collection event report' communication. Our system sends the S6F15 message to request wire bonders to report their states, and wire bonders respond with the S6F16 message, which contains the states and events of wire bonders. The system then parses the received messages and displays the states, wire, and bond head on the screen. Additionally, error messages are displayed to facilitate troubleshooting tasks.

```
14:37:47 Sending Primary Message...
14:37:47     <S1F1 W>
14:37:47 (0) SENT 05 (ENQ)
14:37:47 (0) RECD 04 (EOT)
14:37:47 (0) SENT 0A 00 00 81 01 80 01 00 00 00 28 01 2B
14:37:47 (0) RECD 06 (ACK)
14:37:47 (0) RECD 05 (ENQ)
14:37:47 (0) SENT 04 (EOT)
14:37:47 (0) RECD 0A 80 00 01 02 80 01 00 00 00 28 01 02 41 04 38 30 32 30 41
14:37:47          06 32 38 32 32 33 42 03 C8
14:37:47 (0) SENT 06 (ACK)
14:37:47 Received Reply Message...
14:37:47     <S1F2
14:37:47         <L[2/1]
14:37:47             <A[4/1] "8028">
14:37:47             <A[6/1] "28223B">
14:37:47         >
14:37:47     >
14:37:47 Received Status is...
14:37:47   (Message Sent or Reply Received OK.)
```

Fig. 8. S1F1 and S1F2 message log.

```
16:06:15 Sending Primary Message...
16:06:15     <S6F15 W
16:06:15         <U2[1/1] 0>
16:06:15     >
16:06:15 (0) SENT 05 (ENQ)
16:06:15 (0) RECD 04 (EOT)
16:06:15 (0) SENT 0E 00 00 86 0F 80 01 00 00 00 22 A9 02 00 00 01 E3
16:06:15 (0) RECD 06 (ACK)
16:06:15 (0) RECD 05 (ENQ)
16:06:15 (0) SENT 04 (EOT)
16:06:15 (0) RECD 0E 80 00 06 10 80 01 00 00 00 22 01 03 A9 02 00 00 A9 02 00
16:06:15          00 01 00 02 94
16:06:15 (0) SENT 06 (ACK)
16:06:15 Received Reply Message...
16:06:15     <S6F16
16:06:15         <L[3/1]
16:06:15             <U2[1/1] 0>
16:06:15             <U2[1/1] 0>
16:06:15             <L[0/1]>
16:06:15         >
16:06:15     >
```

Fig. 9. S6F15 and S6F16 message log.

Fig. 10. (Color online) User interface.

Figure 10 illustrates the user interface, which is divided into two parts: the states of wire bonders and the historical area. In the first part, the states of wire bonders are displayed. The three colors (green, yellow, and red) represent the running, warning, and error states, respectively. Each wire bonder is divided into three sections. The first section displays the state of the wire bonder, the second section displays the state of the bond head, and the third section displays the state of the wire. When a yellow or red message appears, an error or warning description is shown below the name of the wire bonder, enabling managers to rapidly address the issues. The second part of the user interface displays the historical data of a wire bonder. For example, in Fig. 10, the historical state records of WB008 in the past hours are shown, and the system automatically calculates the availability as 87.5%. Users can select any wire bonder to view its historical state records and choose the time dimension for display, although the default time interval for requesting states from wire bonders in our system is 5 min. When wire bonders encounter errors, they send S6F5 and S6F11 messages to report their states, and the system updates the states accordingly.

5. Conclusions

In this study, we developed an information system to monitor the states of wire bonders by communicating with SECS sensors. The SECS sensors are composed of the SECS-II and network communication modules. The SECS-II module takes charges of parsing messages from hosts and transmitting to the controller of wire bonders, in order to retrieve the relevant information. The network communication module packs the SECS-II messages into the HSMS format and transmits them over the TCP/IP network. The two modules work tightly to ensure the reliability and applicability of transmitting information between our system and wire bonders. The same system structure can be applied to other equipment for semiconductor production.

The advantages of our system are as follows.

(1) It displays the states and error messages of wire bonders, wire, and bond head for managers to evaluate the situations of wire bonders and efficiently allocate maintenance tasks, thereby reducing the time required for repairing and maintaining wire bonders.

(2) It presents the historical availability of wire bonders, enabling managers to precisely estimate the lifespan of consumables and economically plan their replacement, thereby potentially reducing costs.

HSMS is applied in our system to communicate with wire bonders, which is more convenient and flexible than SECS-I. SECS-I communication requires an interface with a low transmission rate (such as RS-232). The topology of SECS-I demands a hub to connect a limited number of wire bonders, which can result in disorganized and messy cables in the production line. In contrast, HSMS communication only requires wire bonders to connect to a 100BASE-T network. The system and wire bonders simply need to send packets to Ethernet, and TCP/IP ensures the delivery of packets. This communication method simplifies the system design. However, a drawback of HSMS is that it requires wire bonders to support HSMS and be equipped with a 100BASE-T interface. In the near future, as more wire bonders support Wi-Fi connection, the organization of the production line will become neater and tidier.

Unlike the MQTT, OPC UA, and DDS protocols, the current SECS specification does not provide a security mechanism.(8) With the advanced development of software and hardware in the semiconductor industry, more and more equipment will be connected to the Internet. Given that equipment in the semiconductor production line is often expensive and abundant, security issues should receive more attention in the future.

References

1. Y.-J. Park and S. Hur: Processes **8** (2020) 394. https://doi.org/10.3390/pr8040394
2. Y. Long, B. He, W. Cui, Y. Ji, X. Zhuang, and J. Twiefel: Mater. Des. **192** (2020) 108718. https://doi.org/10.1016/j.matdes.2020.108718
3. T. A. Rodrigues, N. Bairrão, F. W. C. Farias, A. Shamsolhodaei, J. Shen, N. Zhou, E. Maawad, N. Schell, T. G. Santos, and J. P. Oliveira: Mater. Des. **213** (2022) 110270. https://doi.org/10.1016/j.matdes.2021.110270
4. O. Vermesan: Artificial Intelligence for Digitising Industry – Applications (River Publishers, New York, 2021) 1st ed., Chap. 2.4.
5. C. C. Kuo, P. C. Chen, and C. T. Tseng: Proc. 2022 Int. Symp. Semiconductor Manufacturing conf. (2022) 1–3.
6. A. Jaisan, S. Manickam, S. Laghari, S. U. Rehman, and S. Karuppayah: Int. J. Adv. Comput. Sci. Appl. **12** (2021) 242. https://doi.org/10.14569/IJACSA.2021.0120828
7. Hume Integration Software: http://www.hume.com/secs/msgs.html (accessed April 2022).
8. S. A. Laghari, S. Manickam, and S. Karuppayah: Int. J. Electr. Electron. Eng. Telecommun. **10** (2021) 105. https://doi.org/10.18178/IJEETC.10.2.105-114