Int J Adv Manuf Technol
DOI 10.1007/s00170-015-7117-z

A novel ISO 6983 interpreter for open architecture CNC systems

Yusri Yusof[1] · **Kamran Latif**[1]

Received: 22 June 2014 / Accepted: 6 April 2015

Abstract Computer numerical control (CNC) technology is a key technology in machine tools and is also the base of industrial unit computerization. CNC machines are operated by controllers, which have a software module inside known as interpreter. The function of an interpreter is to extract data from computer-aided manufacturing (CAM) system-generated code and convert it to controller motion commands. However, with the development of numerical control technology, existing CNC systems are limited with interpreter lacking in expandibility, modularity, and openness. In order to overcome these problems, open architecture technology (OAC) was employed in CNC controller. In this paper, a new technique is presented for the interpretation of the International Standards Organization (ISO) 6983 data interface model. The proposed technique is able to interpret ISO 6983 data and translate it to the internal structure required by the CNC machine. It takes an input file in text format and extracts position, feed rate, tool, spindle, and other data. It is also able to generate output in text and EXtensible Markup Language (XML) files as per user defined file structure. The use of .txt and .xml files enables shop floor data modification and internet accessibility facilities in CNC system. The paper includes an introduction, brief description of the architecture, algorithm design, operational pattern, and validation of the system through experimental studies. The output of these experiments illustrated satisfactory performance of the interpreter with an OAC CNC system.

Keywords Interpreter · Open architecture controller · CAD/CAM · ISO 6983 · LabVIEW

1 Introduction

Over the last 50 years, machine tools have evolved from simple machines to highly sophisticated computer numerical control (CNC) technology which proved to be economical in mass, batch, and many other single-item production cases. The most important factors which contribute towards the economic feasibility of CNC technology are as follows: high productivity rates, uniformity of the product, reduced component rejection, reduced tooling costs, less operator involvement, and easy machining of complex shapes [6]. In the progression towards more modern systems, flexible manufacturing became dominant in 1970s and 1980s to enable low batch production of a wide range of parts. In order to realize flexible manufacturing, computer numerical-controlled machines became a critical manufacturing resource due to their capability for being reprogrammed to produce different parts [7]. Consequently, CNC machines with multi-axis and multi-process workstation configurations were developed to support high-speed manufacturing of precision parts such as complex aerospace components [8].

Since development, CNC machines are operated by International Standards Organization (ISO) 6983 standard formally known as G-M codes. This ISO standard is based on the representation of the tool path digitized with respect

✉ Yusri Yusof
yusri@uthm.edu.my

Kamran Latif
engr.kamranqureshi@gmail.com

[1] Faculty of Mechanical and Manufacturing Engineering, Universiti Tun Hussein Onn Malaysia, Batu Pahat, Malaysia

Published online: 26 April 2015

to a tool size and machine command status [9]. The programs for CNC machines are generated by computer-aided manufacturing systems that utilize CAD information. The G-M codes language is defined by numerical codes such as G, T, M, F, S, etc., indicating the movement of a machine and an axis to the controller. CNC machines are composed of many parts. The controller is one of them and has two module inside: hardware module and software module. The function of the software module is to translate the input code to the internal structure of the machine. CNC controllers read G code (ISO 6983) instructions, interpret them, and perform numerically direct interpolation of the cutting tool in the work piece [9]. Designers generate G code file by using CAD/computer-aided manufacturing (CAM) applications on computers and transfer the code to machine controller for interpretation to execute tool motion control operations [10].

Today's conventional CNC machines are operated by ISO 6983 data interface model, and there are many CAM tools available to generate NC programs. In reality, these G code programs are very valuable because they incorporate both an implicit micro-process plan as well as many years of operator experience [11]. However, when these programs are processed in a CAM system by a machine-specific post-processor, they become machine dependent. Besides that, CNC machine controller vendors have also developed some extensions in G code to add more features in their systems. These extensions vary from vendor to vendor and results in further closeness of controllers [7]. The term closeness means that the controllers act as black boxes. Due to this, it is very difficult for machine tool builders to develop and implement custom control functions [12]. These systems are closed in both hardware and software aspects. In other words, they are closed in terms of Human Machine Interface (HMI) platform, input/output functions, connectivity, data interface model, etc. Hence, it is not surprising that each machine tool manufacturer utilizes their own proprietary hardware and software. As a result, this type of proprietary controller design does not allow the end user to install and interface any new indigenously developed or commercially acquired functional modules autonomously so as to enhance the functionality of the machine. Apart from that, current ISO 6983-based CNC systems transform information in uni-directional flow, which results in no feedback response. Due to this, the shop floor modifications are very hard to perform.

In order to overcome these shortcomings, open architecture technology was introduced into CNC systems. Open CNC controller means "Controllers are independent from manufactures technology, allowing the user to buy hardware and software from several different manufacturers and freely assemble the acquired piece of equipment" [13]. At present, personal computer (PC) as the hardware platform and real-time operating system as the software platform has been the mainstream direction of open CNC systems [14]. This configuration enables the trend of PC-based open soft

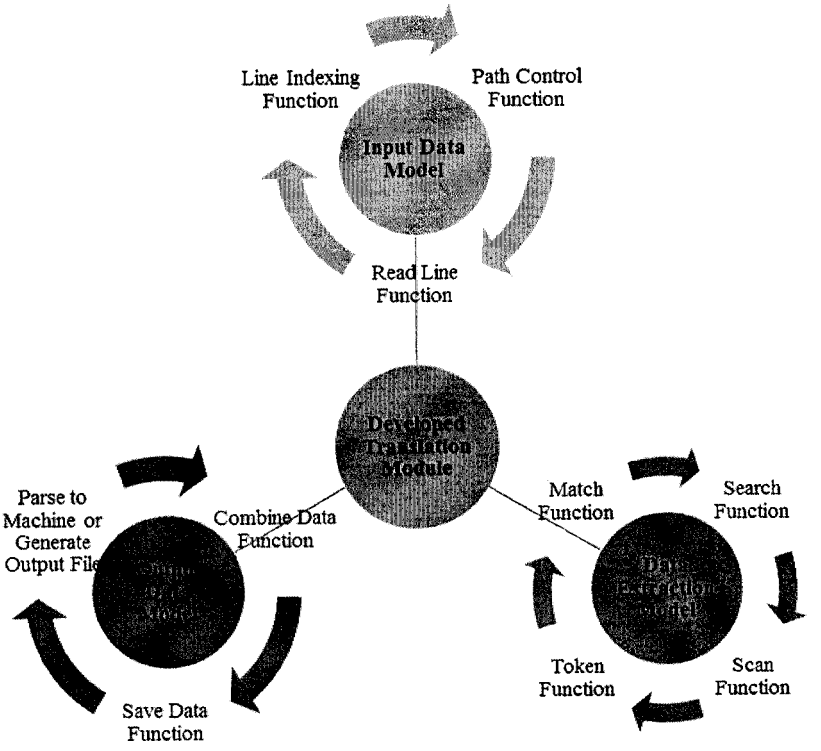

Fig. 1 Internal structure of interpreter

CNC systems [15], which makes the CNC reconfigurable, interoperable, portable, and interchangeable [16].

Based on this technology, there have been some research that are highlighted by different authors such as Xiao and his research cluster introduced an open architecture numerical control system based on Windows CE for turning machine, which decreases processing and memory capacity of the system. But, this system also requires high-end and expensive microprocessors. This system contains a G code interpreter that converts ISO 6983 informations into hoc file [17]. Similarly, Ma along with other researchers developed an Open Modular Architecture Control (OMAC) Application Program Interface (API)-based CNC system for milling machine on VC++ platform. This system includes a G code interpreter for line, arc and non-uniform rational B-spline (NURBS) geometry [15]. Later, Yuan and his research team introduced an OMAC-based CNC system for five-axis CNC machine that is able to interpret linear and spline geometry [18]. After that, Mao and his research cluster proposed an OMAC architecture-based CNC system for three- and five-axis CNC milling machine based on ISO 6983 data interface model [19, 20]. Ekkachai along with his research team proposed an approach to convert commercial CNC system into more open CNC unit. This approach contains a G code interpreter developed by using Visual Studio.Net with C# and C++ platforms [21]. Similarly, Ramesh and Poo introduced an new Ethernet-based three-axis CNC milling system that includes a G code interpreter to translate depth of cut, spindle speed, and feed rate information. This system was developed by using VC++ platform, and data mining algorithm was designed in Matrix Laboratory (MATLAB) by using neural network (NN) technique [22]. Later, Li and Zhang introduced an GRBL G code interpreter that was able to translate only few G code informations [23]. After that, Wang along with other researchers developed an real-time operating system (RTOS)-based embedded CNC system for ISO 6983 data interface model by utilizing Linux RT platform [24]. Similarly, Pabolu and Srinivas introduced an three-dimensional CNC system that was limited only to two-axis machine workability [25]. A major contribution in terms of ISO 6983 interpretation was presented by Liu and his research cluster. They introduced a new technique for the development of universal ISO 6983 interpreter based on C++ platform [26–28]. Later, Xu and his research team developed an ISO 6983 data interface model-based interpreter in C language for B-spline curve information translation [29]. After that, Chen and his research cluster proposed an new ISO 6983 interpreter designed in C++ for commercial and PC-based CNC controller with tool path planning module to improve the efficiency [30]. Similarly, Khana along with other researchers developed a low-cost production system based on Arduino technology for turning machine that converts G code into NIST-SAI Conical Code (NCC) [31].

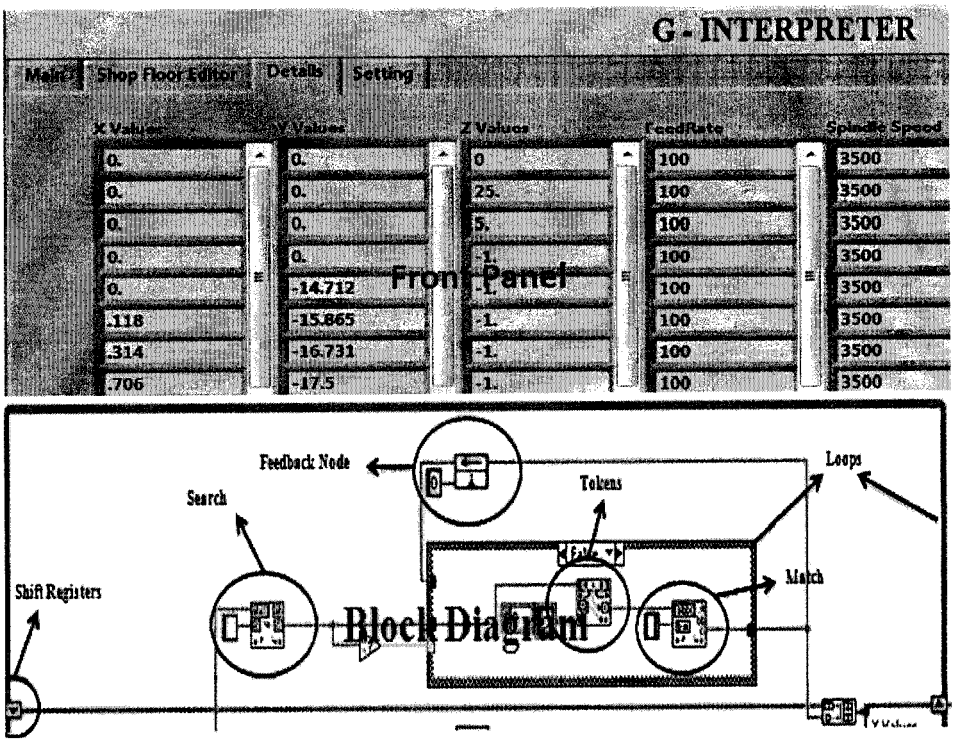

Fig. 2 Data extraction module

Obtaining idea from those previous efforts, in this study, a new technique for the interpretation of ISO 6983 data interface model has been presented. The proposed technique is based on the National Instruments (NI) Laboratory Virtual Instrument Engineering Workbench (LabVIEW). In past, not many had utilized NI LabVIEW platform for ISO 6983 interpretation; most of the previous approaches were based on C, C++, JAVA, VB, etc. platform. However, some had utilized LabVIEW platform for CNC motion control techniques such as[32–34]. However, these tecniques does not contain any interpretation module for ISO data interface models. Whereas proposed technique can be integrated with those approaches. The proposed technique also provides feedback of data and enables a minute part of shop floor data modifications into ISO 6983-based CNC systems. Apart from that, proposed technique also enables internet accessibility into open CNC system by generating .xml and .txt physical file outputs in user-defined file structure. However, commercial CNC systems are not able to provide all these facilities. Other than that, another main objective of the proposed system is to open the doors for other contributors for enabling of various functionalities in CNC systems such as monitoring, inspection, simulation, etc.

Overall, in this paper, a novel ISO 6983 interpreter for open architecture CNC control system has been presented. Development of an ISO 6983 data interface model-based interpretation module for open architecture controllers is discussed in Section 2. In Section 3, algorithm design is discussed in details. The working principle of the developed interpreter is addressed in Section 4. Section 5 highlights the experimental study performed by using the developed interpreter on an open architecture control CNC machine. Finally, some discussion, future suggestions, and conclusions on the developed interpreter are presented in Section 6.

2 Architecture of interpreter

The developed interpreter is a software component to translate existing CAD/CAM system-generated ISO 6983 code into an open architecture-based CNC system understandable data structure. It has been programmed in NI LabVIEW. The LabVIEW platform was chosen because it is a graphical language; therefore, it is easy to use as compared to text coding platforms. It provides a wide range of connectivity through serial ports, wireless, etc. [35–37]. Another advantage of LabVIEW is that it provides connectivity access to almost all the software such as MS office, C, C++, Java, Matlab, Solidworks, etc. These facilities of LabVIEW helps to provide an open environment to CNC, that can perform various functions such as monitoring, inspection, interpretation, simulation, database connectivity, and internet connectivity in a single unit. The function of the developed interpreter is to translate G code commands, so that the stepper/servo motors can be moved using linear or

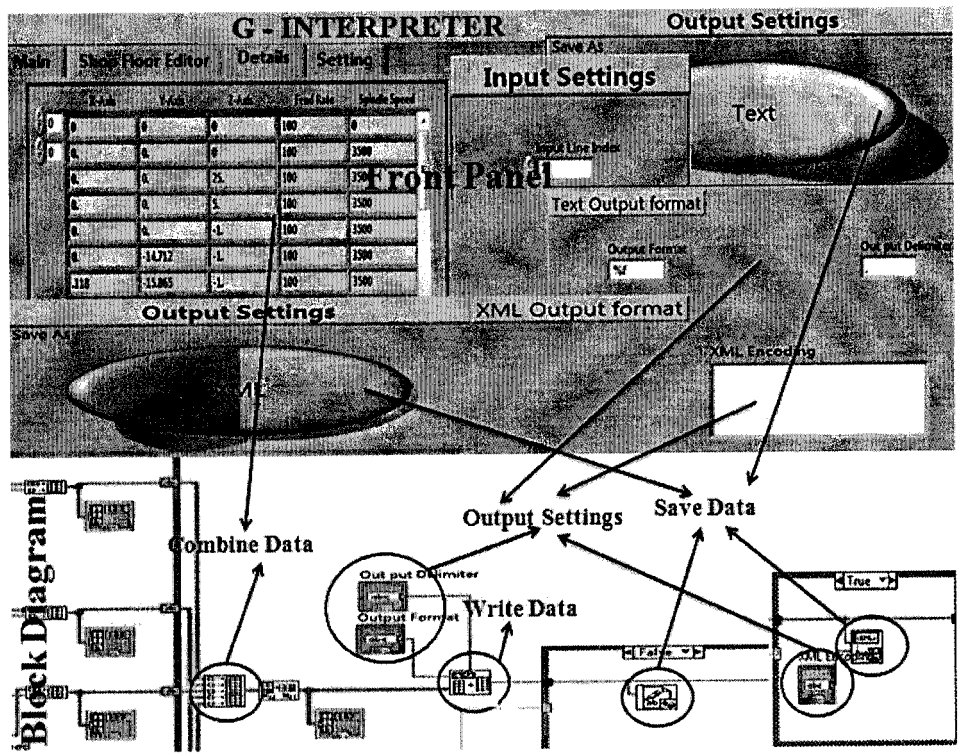

Fig. 3 Output data module

circular interpolation. It interprets the acceleration, deceleration, spindle speed, feed rate, and other information for each stepper/servo motor. The interpreter reads ISO 6983 code line by line in text format and extracts position, feed rate, spindle, tool, and other data. The internal structure of the interpreter is composed of three modules: *input data*, *data extraction*, and *output data*. Each module contains various functional blocks as shown in Fig. 1.

2.1 Input data module

This module is composed of *file path*, *read text*, and *line indexing* functional blocks. The role of this module is to provide a path control for file uploading, read complete content of the input file, and provide guidelines to the interpreter regarding reading of the input file. This module is the initial phase of the developed interpreter, whose functionality initiates with the *file path* functional block that provides a path control to get the input source code. The *read text* functional block is responsible to read complete content of the input source code. Then, at the end of this phase, *line indexing* functional block is developed. The purpose of this functional block is to guide the interpreter regarding the starting point of the code. With the enabling of this function, the interpreter is able to extract/read code from any line of input file.

2.2 Data extraction module

This module is composed of *search*, *scan*, *tokens*, and *match* functional blocks. It is the combination of both lexical scan and syntax analysis functions. The module begins with a search for patterns followed by a scan for pattern data, generation of tokens, and matching of regular expression functions. In this phase, an algorithm for data mining is developed. The development of algorithms initially starts with the search for the patterns in each line of input code such as X, Y, Z, F, S, and others. The output of this functional block is in the form of divided sub-codes. After dividing the input into sub-codes, the *scan* functional block start that scans the input file for the values of patterns sought. After that, the *token* functional block is executed, which breaks the scan data into token strings. The last function of this phase is to search for regular expressions in the token strings. The output of this search is the extracted values of position, feed rate, spindle, and other data. In order to extract complete data from input code, the algorithm repeats all mentioned processes by utilizing loops, structures, feedback nodes, shift register, etc. functions. With the help of these functions, the algorithm extracts complete data sets from the input file. Figure 2 shows the front panel and block diagram of the *data extraction module*.

2.3 Output data module

The function of the *output data module* is to combine extracted data, write combined code, and pass written data to an open CNC machine. This module also saves the data into text and xml file in a user-defined structure. These files enables feedback of information and web-based facilities in the proposed system. This module is composed of *combine data*, *write data*, *save data*, and *output file format* functional blocks. Firstly, it combines all the extracted data into single code by utilizing *combine data* functional block. This combined code works as the shop floor editor, which provides

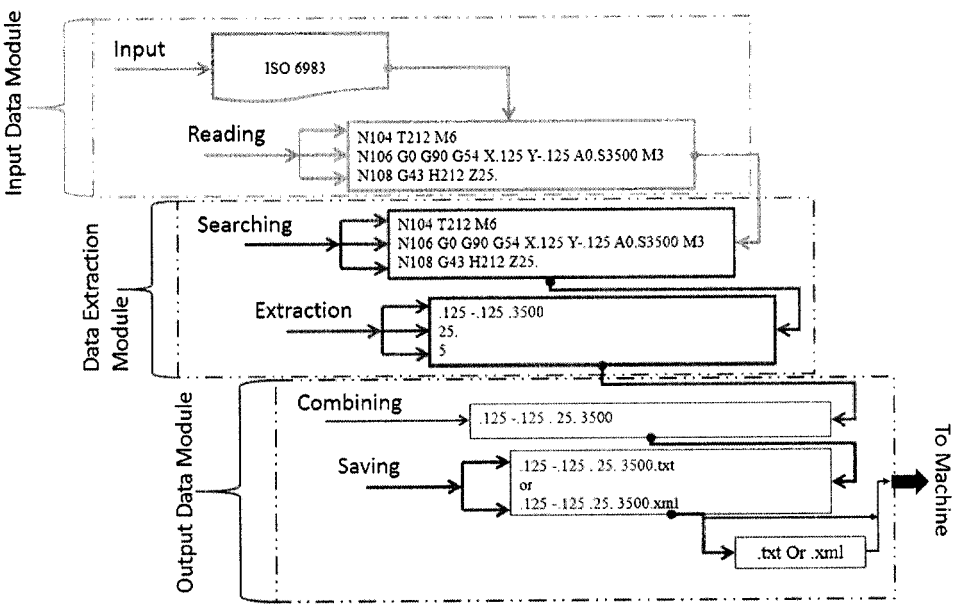

Fig. 4 Algorithm design of interpreter

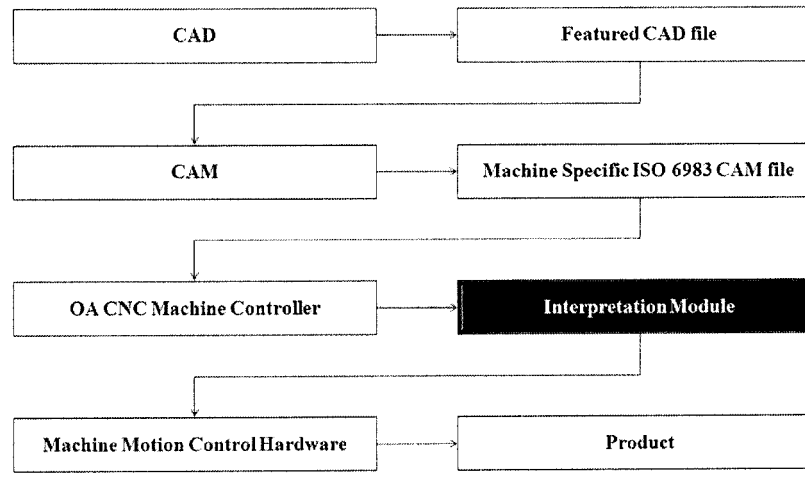

Fig. 5 Working cycle of developed interpreter

access to last minute corrections and modifications to data on the shop floor. Then, the combined code is passed to the CNC machine controller hardware module in a user-defined file format (.txt or .xml) by *output file format* functional block in a correct way, e.g., axis values go to the relative servo/stepper motors, tool values go to the tool changer. At the same time, this functional block is also able to generate output files in .txt and .xml formats into a user-defined file structure. The generated output files can be used on any type of PC-based open architecture CNC machine system. Figure 3 shows the front panel and block diagram of the output data module.

3 Algorithm design

An algorithm has been designed in order to execute developed modules as per required structure of operations. That starts with the *path control* functional block, which provides a path for an input file to be loaded into the interpreter. After that, the *read data* functional block starts, which reads the complete contents of the input file. The algorithm makes a decision at this stage regarding input data, if input file does not contain ISO 6983 information, it passes the signals to error msg. However, in other case, the algorithm passes the data to the *line index* functional block that guides the

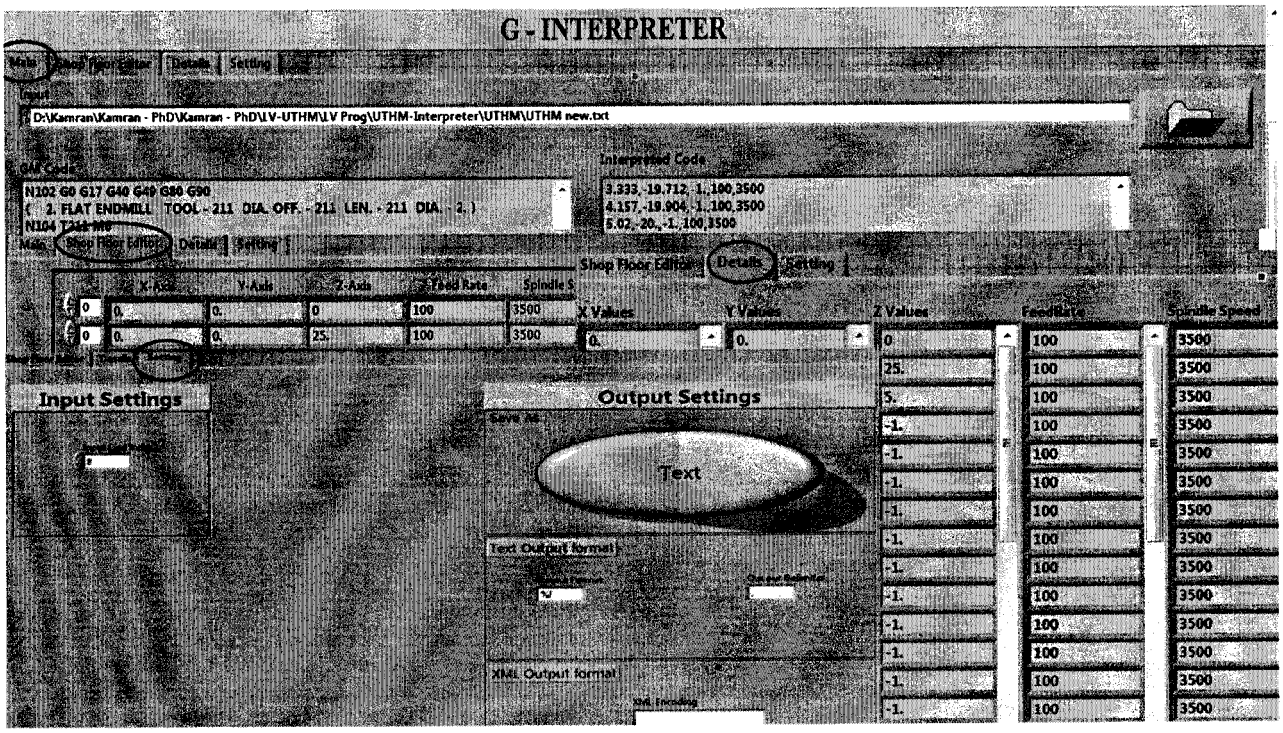

Fig. 6 GUI of interpreter

interpreter regarding the starting point of file reading. By default, it has an index value equal to zero that means the first line of input code. After that, the *data extraction* module is activated with *search* functional block, which searches for patterns (X, Y, Z, F, S, T, etc.) inside the input code. At this moment, the algorithm makes a decision regarding patterns. If no pattern is found, the algorithm shows an error message. If a pattern is found, the algorithm passes the data to *scan*, *token*, and *match* functional blocks step by step. The *scan* functional block scans the input file for pattern values, then these values are converted into *token* strings by the token functional block. After that, the token strings are passed to the *match* functional block, which extracts the values of patterns from token strings. Then in last, *output data* module is executed, which first combines all the extracted data of previous modules and saves it into internal memory by utilizing *combine data* and *save data* functional blocks. Then, the *output* functional block starts that passes the data directly to a CNC machine and also generates physical files in .txt and .xml formats as per user-defined structure as shown in Fig. 4.

4 Operational pattern

The developed interpreter works with currently available CAD/CAM systems. Its working cycle starts with the CAD design, which utilizes CAM software for the generation of ISO 6983 code. The proposed interpretation technique takes that machine specific CAM file in .txt format as input and extracts the position, spindle speed, feed rate, tool, and other data from input code. After the extraction, the interpreter generates output in user-defined .txt and .xml files. The interpreted data file is then transferred to the open architecture CNC machine, which perform operations as shown in Fig. 5.

The graphical user interface (GUI) of the developed interpreter contains four tabs: *main*, *shop floor editor*, *details*, and *settings* as shown in Fig. 6. The *main* page of the GUI shows the input file path control, input ISO 6983 code and translated code. The *shop floor editor* shows all the extracted data as a single code so that the user can easily modify data at any time while working or off-line. The *details* tab shows the extracted values into separate columns

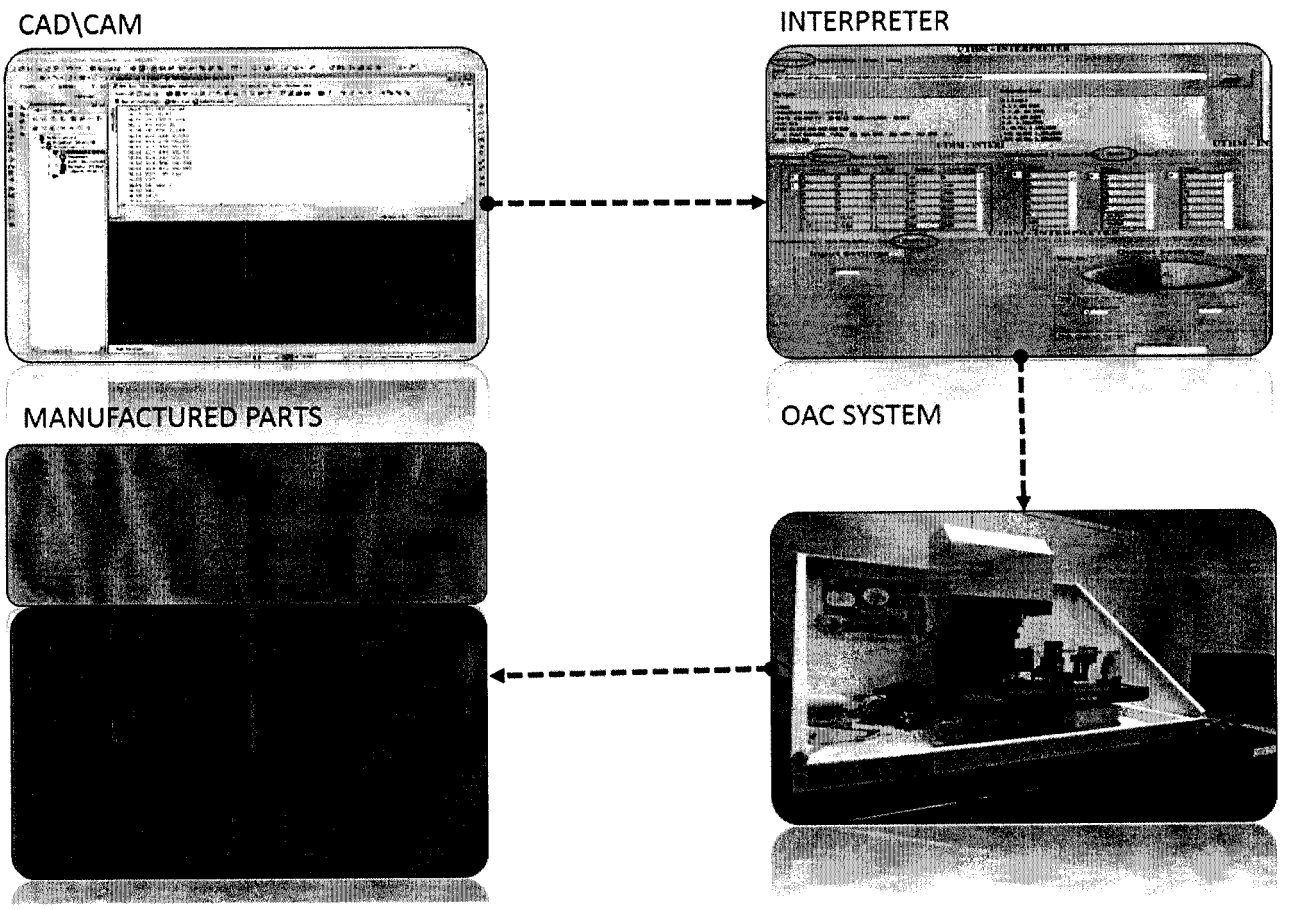

Fig. 7 Experimental setup of developed interpreter

Table 1 Summary of commercial and proposed interpreter features

Interpreter	Nature	HMI Platform	Input / Output Files	Connectivity	Data Interface Model	Feed back
Fanuc 30i, 35i, 0i	Close	PC with WinCE/NT	.NC	LAN, USB, Floppy	ISO 6983	No
Indramat	Close	PC with Win NT	.NC Floppy	LAN, USB,	ISO 6983	No
MDSI	Close/open	PC with Win NT	.NC LAN, USB, Floppy	ISO 6983	No	
Robert Bosch Anger HCK, Orthodyne 360c	Close NT	PC with Win	.NC Floppy	LAN, USB,	ISO 6983	No
Allen Bradely Series9, 8200AB	Close	PC with Win NT	.NC	LAN, USB, Floppy	ISO 6983	No
Siemens H250, 808c, MCP 483c	Close/ Open	PC with Win 95 to Xp/NT	.NC	LAN, USB, Floppy	ISO 6983	No
Mazak 430 A, 410A	Close	PC with Win 95 to Xp/NT	.NC	LAN, USB, Floppy	ISO 6983	No
Denford NS, ATC NS	Close	PC with Win 95 to Xp/NT	.NC	LAN, USB, Floppy	ISO 6983	No
Developed Model	Open	PC with Win 95 to 8/ NT/ Linux	.txt and .xml	LAN, USB, Wireless, Floppy, Internrt	ISO 6983 and ISO 14649	Yes

for easy understanding. The *setting* tab is the important tab of the developed interpreter, which is composed of input and output settings. The input setting is to guide the interpreter regarding the starting point (the point from where the interpreter starts reading the input code). By default, this has a value of 0 that indicates first line of the input code. Output settings have three steps: the first step is for selecting the output file format (.txt or .xml). The second step is for format and delimiter settings of text file format. The third step is for .xml output settings using XML encoding controller.

5 Validation of interpreter

In this study, various experiments have been performed for validation of the developed interpreter. These experiments start with the design process by using CAD/CAM software. After designing, CAM machine features are given to the designs and ISO 6983 codes for each design are generated. Then, CAD/CAM generated codes are saved in .txt format by using CAM software code editor. After the generation of ISO 6983 codes, the next step is to upload the codes into the developed interpreter. Before performing the interpretation operation, the setting of the output file is carried out as per requirements from the *setting* tab. In this experiment, the output code is formatted as floating point number with comma (,) delimitation. After all settings, the code is executed and the output file is saved in .txt format. Finally, the translated codes are transferred to an open architecture control CNC machine [38] and operations are carried out. Figure 7 shows the experimental setup and manufactured parts.

6 Discussion and conclusion

Conventional CNC machines are operated by controllers. Each controller has a software module inside known as an interpreter. The function of the interpreter is to translate input ISO data interface model information into a required internal data structure of CNC machine. As these CNC controllers are closed in nature, therefore, they are dependent on vendor specifications. Due to this, interpreter is also of closed nature. In order to overcome these shortcomings, an open architecture CNC control systems was introduced. In this work, a new technique for ISO 6983 data interface model interpretation with shop floor modification and internet accessibility has been introduced. The developed interpreter was programmed in National Instruments LabVIEW and is independent of vendor specifications. It is open in nature, HMI platform, input/output function, connectivity, etc. It can be used anywhere without any additional hardware requirements. Table 1 shows the summary

26. Liu Y, Guo X, Li W, Yamazaki K, Kashihara K, Fujishima M (2007) An intelligent NC program processor for CNC system of machine tool. Robot Comput Integr Manuf 23(2):160–169
27. Guo X, Liu Y, Yamazaki K, Kashihara K, Fujishima M (2008) A study of a universal NC program processor for a CNC system. Int J Adv Manuf Technol 36(7-8):738–745
28. Guo X, Liu Y, Du D, Yamazaki K, Fujishima M (2012) A universal NC program processor design and prototype implementation for CNC systems. Int J Adv Manuf Technol 60(5-8):561–575
29. Xu X, Li Y, Sunn J, Wang S (2012) Research and development of open CNC system based on PC and motion controller 29: 1845–1850
30. Chen L, Yu D, Zhang H, Geng C, Dong L (2012) Design implement of a modularized CNC interpreter based on the integration of tool path planning module. Computer Science and Automation Engineering (CSAE 2012). International Conference on (IEEE 2012). 613–616
31. Khanna A, Kumar A, Bhatnagar A, Tyagi R, Srivastava S (2013) Low-cost production CNC system. Intelligent Systems and Control (ISCO), 2013 7th International Conference on (IEEE 2013): 523–528
32. Zhanbiao G (2010) An open CNC controller based on LabVIEW software. 2010 International Conference on Computer Application and System Modeling (ICCASM 2010) 4:
33. Weidong Y, Zhanbiao G (2010) An open CNC controller based on LabVIEW software. Computer Application and System Modeling (ICCASM), 2010 International Conference on (IEEE 2010)4: 476–479
34. da Rocha P, Diogne de Silva e Souza R, de Lima Tostes ME (2010) Prototype CNC machine design. Industry Applications (INDUSCON), 2010 9th IEEE/IAS International Conference on (IEEE 2010): 1–5
35. Bishop RH (2009) LabVIEW 2009 student edition. Prentice Hall Press
36. LabVIEW F (2009) National Instruments. Austin, Texas:78730–5039
37. Elliott C, Vijayakumar V, Zink W, Hansen R (2007) National instruments LabVIEW: a programming environment for laboratory automation and measurement. J Assoc Lab Autom 12(1):17–24
38. Yusof Y, Latif K (2013) Frame work of LV-UTHM: an ISO 14649 based open control system for CNC milling machine. Applied Mechanics and Materials 330:619–623