Journal of Engineering and Applied Sciences 11 (Special Issue 2): 3273-3278, 2016
ISSN: 1816-949X
© Medwell Journals, 2016

FPGA Hardware Implementation for Accelerating QR Decoding

Muhammad Alhammami, Chee Pun Ooi, Wooi-Haw Tan and Soon Nyeancheong
A Faculty of Engineering, Multimedia University, Jalan Multimedia, Malaysiarul Ehsan,
63100 Cyberjaya, Selangor, Malaysia

Abstract: QR codes has gained more attention as an input interface to many embedded applications. However, some applications need extra computing resources which demandhigh-performance QR code decoder. This study suggests a hardware solution to accelerate the decoding function. The proposed design is implemented using CYCLON II FPGA from Altera with the decoded results display on a LCD. The initial experiments show that it is possible to decode the unmasked QR raw bits efficiently in real time which shows good potential to offload the computationally intensive task of QR image decoding process from the main processor and to room for advanced image pre-processing and security decryption algorithm to be implemented in FPGA.

Key words: QR code, QR decoding, FPGA, hardware implementation, demandhigh-performance

INTRODUCTION

The intensive visual and informative features make Quick Response codes, QR an indispensable visual input method in many embedded systems applications (Alapetite, 2010; Canadi *et al*., 2010; Al-Khalifa, 2008; Huang *et al*., 2010). These applications include manufacturing process where products and equipment could be trace for process and production management. QR based product traceability in warehouses and logistics industry. In thehealthcare sector, QR is used for patient IDs, medication tracking and specimens tracking. QR is also widely used in thetransportation industry for ticketing and boarding passes. The most popular usage of QR codes is in the marketing industry for web uniform resource identifier, URL links, mobile marketing, e-Coupons, payments, etc.

In this study we present a hardware solution to decode the unmasked QR raw bits in real time with a 5M CMOS sensor to capture the QR image. The proposed design realised by a CYCLON II FPGA on DE2 board which has an LCD on board to display the decoded message.

Literature review

QR structure: The structure of QR code is in black and white checkered pixel patterns divided to many specific regions (Lisa and Piersantelli, 2008). Each region has a specific purpose. An example of a version 3 QR code is shown in Fig. 1 to illustrate the structure of a QR code:

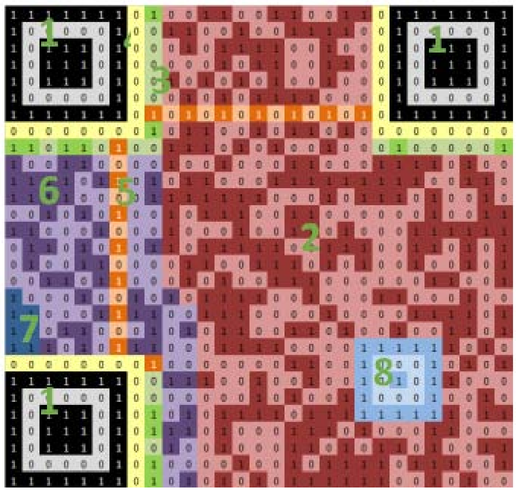


Fig. 1: The QR structure

- Finder pattern/positioning marking (black regions); three identical regions located in three corners of the QR Code. Each pattern represented by a N×N matrix. Finder Patterns guide the decoder to detect the QR Code and determine the correct orientation
- Data (pink regions), blocks of bits which contain the actual data
- Format Information (green regions); these bits contain information about the error correction level and the masking pattern used
- Separators (yellow regions); these regions are white which means they contains only zero bits. The width

of each white separator is one pixel. Separators regions improve the recognizability of the finder patters by separating it from the actual data

- Timing Pattern (orange regions); alternating black and white bits helps in determining the width of a single square
- Error correction (purple regions); the number of error correction words is determined according to the version of the code and its error level
- Remainder bits (dark blue); this region consists of empty bits but not zero by necessity, comes after the error correction region
- Alignment patterns (bright blue); used by thedecoder to compensate for moderate image distortions. Depending on the version, these patterns may be more than one or zero
- The QR code is surround by the quite region, (aregion with all zero bits) to distinguish the QR Code from its surroundings

Reading QR code: QR codes have a complex reading process so it requires long computation time. However, this processing time is still acceptable for many applications. All existing QR decoder implemented are software based, either in mobile phone apps or other microprocessors based embedded systems such as Raspberry Pi. Nevertheless for more advanced applications such as data decryption for data security where extra real-time and complex functions will occupy most of the computational resources of the processor. Thus using Field Programmable Gate Array (FPGA) as a co-processor to offload and accelerate the QR code processes from the main processor in embedded systems sounds promising in increasing the performance of the whole systems.

The QR decoding process is divided into three main phases, namely image acquisition, image pre-processing and QR decoding. The computation task involved for each of this phase is depicted clearly in Fig. 2.

Related work: Up to our knowledge, there is no previous contribution in designing QR hardware decoder using FPGA. However, there are few contributions of using FPGA in generating the QR codes (Liu and Liu, 2006) and enhancing the QR image pre-processing technique (Kinjal and Hiren, 2014). More contributions exist in software-based decoders using existing function stored in the libraries. From other literature we can conclude that the use of FPGA in QR decoding process has not been investigated.

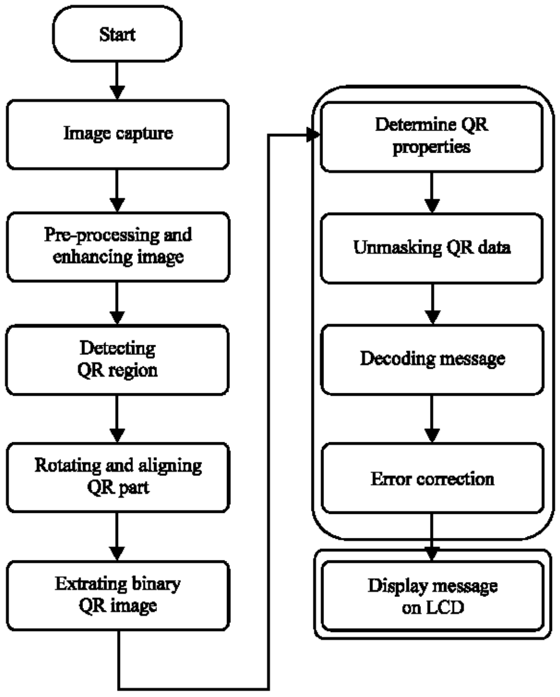


Fig. 2: Flow chart of reading QR code

MATERIALS AND METHODS

System design: The input of our proposed design is the unmasked binary QR image stored in a vector scheme like the real unmasked binary image in Fig. 3. The proposed design consists of five steps:

- The unmasked data words are stored in a ROM when the system start
- After getting a start command, mode indicator bits are read and analysed
- Corresponding message bits are fetched from the ROM
- The message of this segment is decoded and shown on the screen
- This operation is repeating until the end mode

Fetching data from rom block: This block serves for fetching series of bits from the unmasked QR image code stored in the ROM. The main inputs of this block are the system clock, address bus where the current segment starts, thenumber of required bits and reading command. The main outputs are the required bits and data ready signal for synchronisation aspect. The block consists mainly of thetimer, comparators and shift registers.

Mode indicator blocks: Whenever the data ready signal from the fetching data ROM block is triggered this block

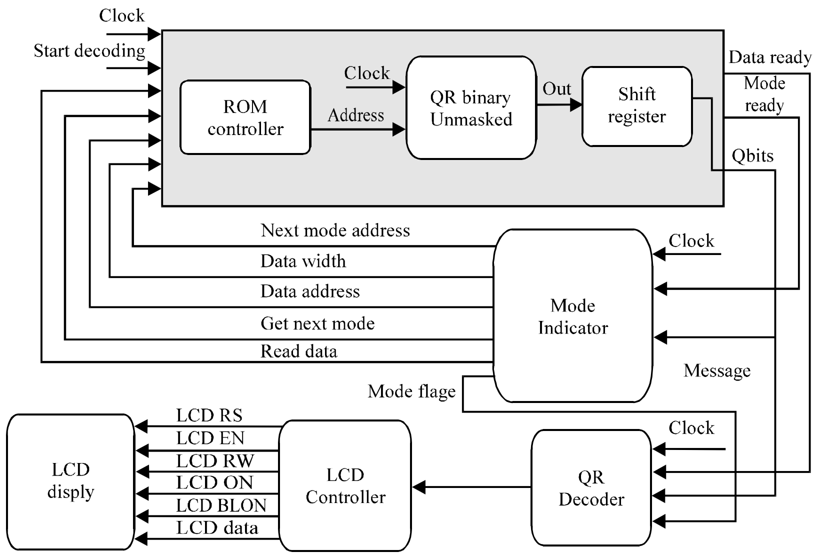


Fig. 3: Block diagram of proposed design

Table 1: Mode indicator

Mode	Indicator
Numeric mode	0001
Alphanumeric mode	0010
8-bit byte mode	0100
Kanji mode	1000
Structured append	0011
FNC1 (first position)	0101
FNC1 (second position)	1001
Terminator/end	0000

Table 2: Length of character count indicator

Version	Numeric	Alpha numeric	8-bit	Kanji
1-9	10	9	8	8
10-26	12	11	16	10
27-40	14	13	16	12

will read the mode bits (4 bits per mode) and the length of the characters. The various mode indicator bits are listed in Table 1. The most common modes used in applications are:

- Numeric mode which encodes only numbers
- Alphanumeric mode which includes only upper case letters only, numbers and few more characters like $, +, -, % and whitespace
- Byte or 8-bit mode which able to encode ASCII characters
- Kanji characters encoding mode

To know the length of data characters, this block examines the bits right after the mode indicator and then depending on the mode and the version of the QR. (Table 2) shows the width of length field for each version and mode as specified in QR standards. The block initializes the corresponding mode of the current segment, length of data, the address of the words of this segment, the number of reading operations needed from the ROM to get the whole segments. The address of the mode bits of next data segment.

Table 3: Compiling summary

Family	Cyclone II
Device	EP2C35F672C6
Timing models	Final
Total logic elements	5,633/33,216 (17%)
Total combinational functions	5,404/33,216 (16%)
Dedicated logic registers	971/33,216 (3%)
Total registers	971
Total pins	270/475 (57 %)
Total virtual pins	0
Total memory bits	880/483,840 (<1%)
Embedded multiplier 9-bit elements	0/70 (0)
Total PLLs	1/4 (25)

QR decoder block: This block decodes the current segment of data part based on the mode flags coming from the mode indicator block which indicates the type of the current mode, whenever the data ready signal is triggered. The data region may consist of many segments with different encoding modes. Each segment has its mode and length of data (Fig. 4). Table 3 shows how multiple modes can be mixed in a single QR code Lisa and Piersantelli (2008). A VHDL function is created in this project to decode each mode using lookup tables and some division components. At this stage, only three modes decoding are implemented which are abyte, alphanumeric and numeric modes.

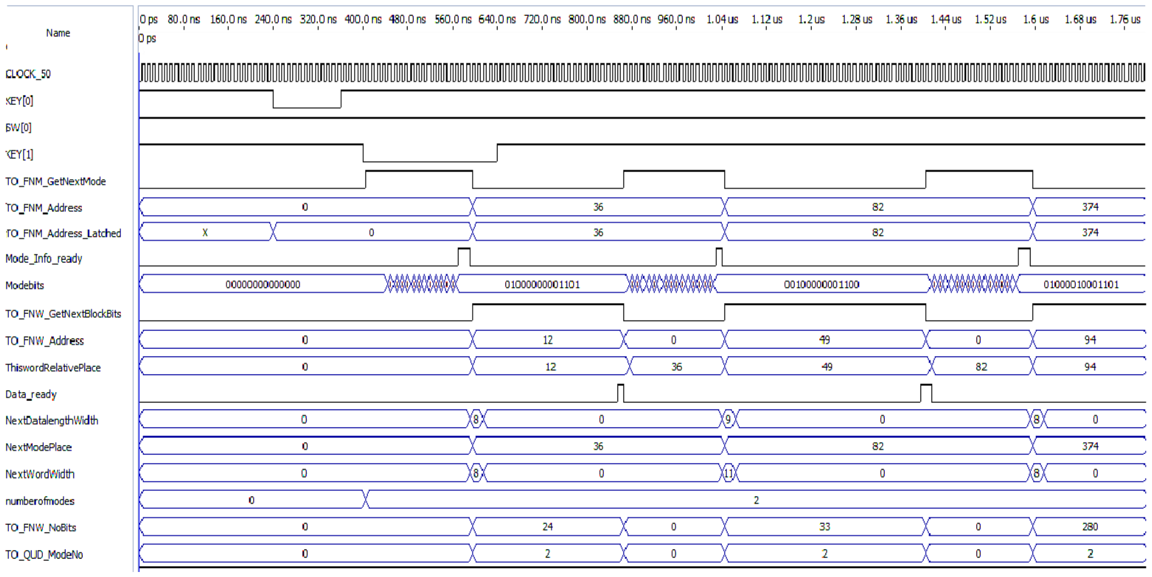


Fig. 4: Functional simulation-A

Message containing several segment

Seg 1:

- Made indicator
- Character count indicator
- Data

Seg 2:

- Made indicator
- Character count indicator
- Data

Seg n:

- Made indicator
- Character count indicator
- Data

RESULTS AND DISCUSSION

The proposed design is realised on a Cyclone 2 FPGA chip mounted on DE2 board. This board has an onboard LCD which we use it to display the decoded message. The compilation summary listed in Table 3 shows that only 17% of the total logic elements, 1% of memory and 3% of the dedicated registers is utilised that leave plenty of room for future development.

Functional simulation: Figure 5 shows the functional simulation of the proposed design where all main signals and other signals are included. The decoded QR code extracted from the sample QR code (Fig. 6) under test has four modes: Byte, Alphanumeric, Byte and the End or Terminate mode. The addresses of the mode are (0, 36, 82, 374) respectively as designated by “TO-FNM-Address” shown in Fig. 5. The addresses of the data words are (12, 49, 94) respectively as indicated by “TO-FNW-Address”. The two signals “Mode-Info-Ready” and “Data-Ready” are status signals to trigger the ROM Block.

Figure 6 shows more detailed zoom for decoding the first mode of the sample QR code under test (Fig. 7). The first mode address is “0”. When the mode indicator block finishes reading and analysing the mode, it raises up the signal “Mode-Info-Ready” at point “A”. The Byte-flag is triggered at point “B” to indicate this mode is Byte mode. The words of this mode sections start at the address “12” as shown in “TO-FNW-Address”. The decoder block reads these data bits from the ROM and decodes it based on the condition specified in QR standard for byte mode. “Data-Ready” signal is asserted at point “C” whenever the decoding is completed and the message is put on its output. The output in this case is characters “J”, “I”, “s” (point D) which tally with the first three characters shown in Fig. 7. The address of next mode section is determined by the signal “NextModePlace” which is 36 in this case.

Experimental results: An arbitrary sample QR code as shown in Fig 7 is employed to verify the functionality of the proposed decoder. The decoded characters shown on the LCD verified the correctness of the decoding algorithm coded in VHDL. The LCD size is two lines with 16 characters; the system will overwrite previous characters and only the last 32 decoded characters are

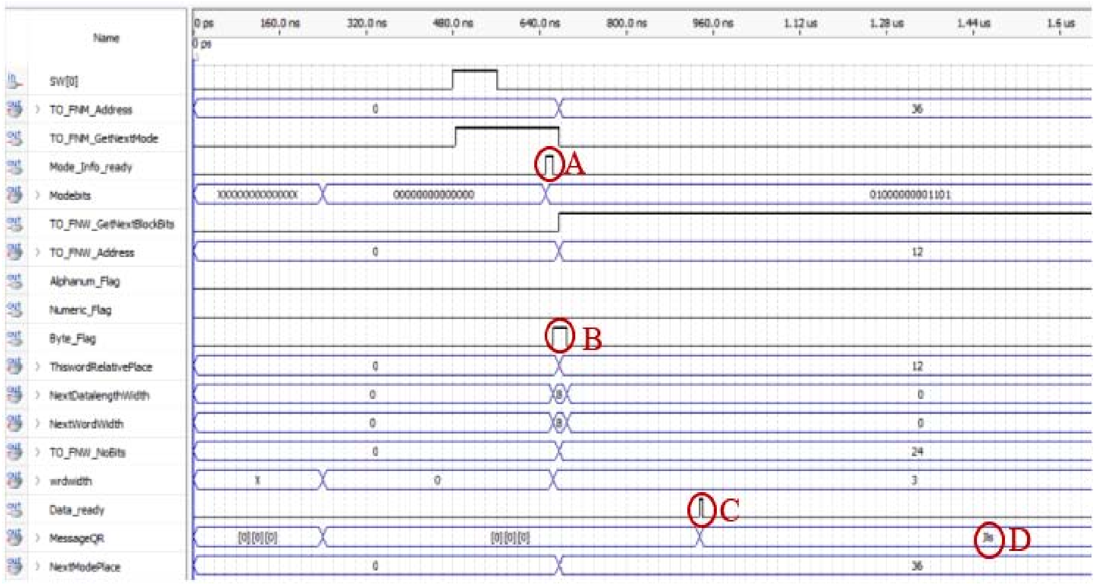


Fig. 5: Funtion simulation-B

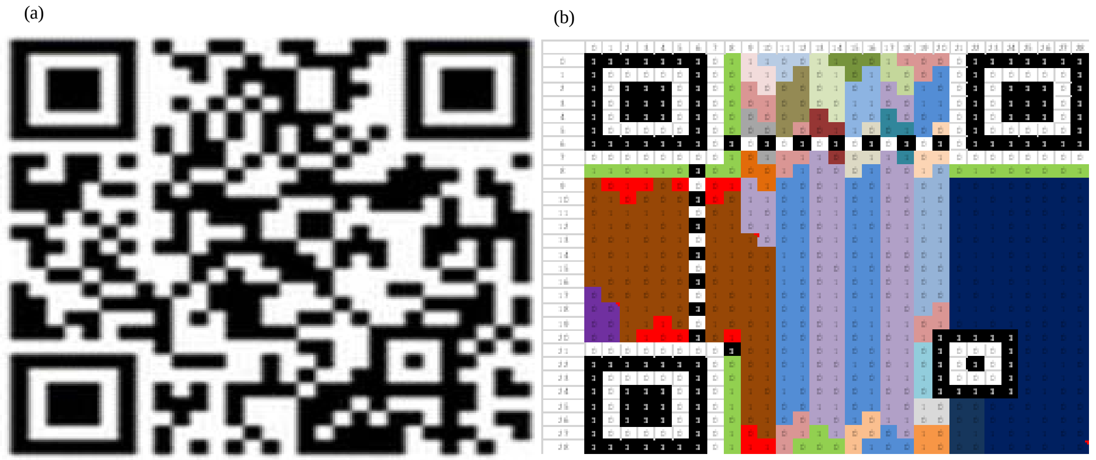
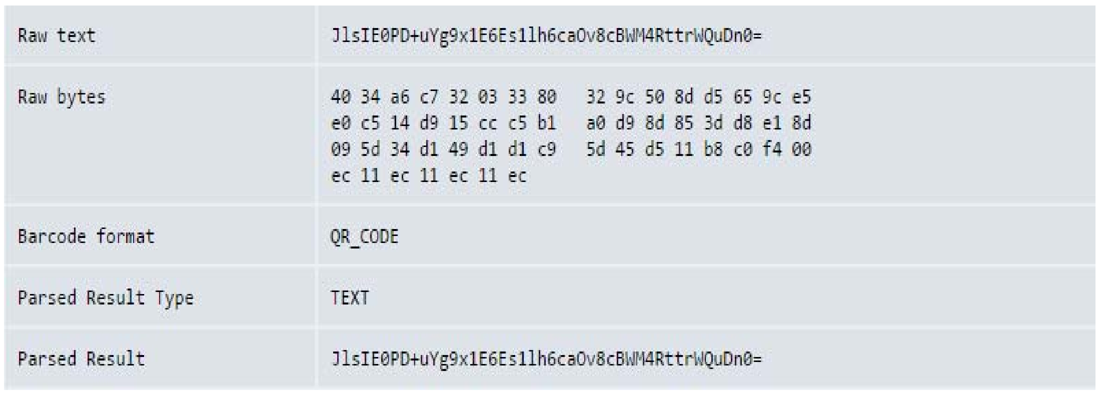
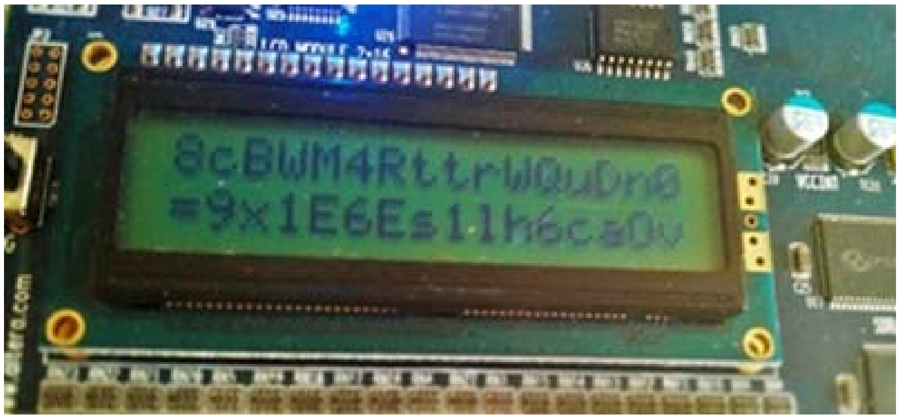


Fig. 6: Arbitrary QR example to be decoded: a) An example of QR code and b) Corresponding unmasked binary image



Raw text	JlsIE0PD+uYg9x1E6Es1lh6caOv8cBWM4RttrWQuDn0=
Raw bytes	40 34 a6 c7 32 03 33 80 32 9c 50 8d d5 65 9c e5 e0 c5 14 d9 15 cc c5 b1 a0 d9 8d 85 3d d8 e1 8d 09 5d 34 d1 49 d1 d1 c9 5d 45 d5 11 b8 c0 f4 00 ec 11 ec 11 ec 11 ec
Barcode format	QR_CODE
Parsed Result Type	TEXT
Parsed Result	JlsIE0PD+uYg9x1E6Es1lh6caOv8cBWM4RttrWQuDn0=

Fig. 7: The decoded characters of the example using a commercial software



```
8cBWM4RttrWQuDn0
=9x1E6Es11h6caDv
```

Fig. 8: The last 32 decoded characters

display in the LCD. Figure 7 shows a snapshot of the decoded result. In comparison between Fig. 7 and 8, we see the correctness of the results.

CONCLUSION

The hardware realisation of QR decoder starting from an unmasked raw data is presented in this study. All necessary blocks are designed in VHDL and implemented using A CYCLON II FPGA chip mounted on the DE2 board. The resources used for this decoder is very small (i.e., 17% of LE) and performance is in real time. Decoded output message is displayed on the LCD. The result of this work shows good potential to offload the computationally intensive task of QR image decoding process from the main processor and room for advanced image pre-processing and security decryption algorithm to be implemented in FPGA. Our next step is to design the stages from capturing the QR image then finding, extracting and unmasking the QR code. The output of these stages will be the input of the system presented in this study.

ACKNOWLEDGEMENTS

The researchers gratefully acknowledge the financial support from the Telekom Malaysia research and development fund (MMUE/150004) and multimedia university (Cyberjaya campus) to implement the project successfully.

REFERENCES

Al-Khalifa, H.S., 2008. Utilizing QR Code and Mobile Phones for Blinds and Visually Impaired People. In: Computers Helping People with Special Needs, Miesenberger, K., J. Klaus, W. Zagler and A. Karshmer (Eds.). Springer, Berlin, Germany, ISBN:978-3-540-70539-0, pp: 1065-1069.

Alapetite, A., 2010. Dynamic 2D-barcodes for multi-device Web session migration including mobile phones. Pers. Ubiquitous Comput., 14: 45-52.

Canadi, M., W. Hopken and M. Fuchs, 2010. Application of QR Codes in Online Travel Distribution. In: Information and Communication Technologies in Tourism 2010, Gretzel, U., R. Law and M. Fuchs (Eds.). Springer, Berlin, Germany, ISBN:978-3-211-99406-1, pp: 137-148.

Huang, Y.P., Y.T. Chang and F.E. Sandnes, 2010. Ubiquitous information transfer across different platforms by Qr codes. J. Mob. Multimedia, 6: 3-13.

Kinjal, H.P. and G.J. Hiren, 2014. A survey on QR codes: In context of research and application. Intl. J. Emerging Technol. Adv. Eng., 4: 258-262.

Lisa, S. and G. Piersantelli, 2008. Use of 2d barcode to access multimedia content and the web from a mobile handset. Proceedings of the Conference on Global Telecommunications, November 30-December 4, 2008, IEEE, New York, USA., ISBN:978-1-4244-2324-8, pp: 1-3.

Liu, Y. and M. Liu, 2006. Automatic recognition algorithm of quick response code based on embedded system. Proceedings of the 6th International Conference on Intelligent Systems Design and Applications, October 16-18, 2006, IEEE, New York, USA., ISBN:0-7695-2528-8, pp: 783-788.

Narayanan, A.S., 2012. QR codes and security solutions. Intl. J. Comput. Sci. Telecommun., 3: 69-71.