FPGA Implementation and Verification of Reed-Solomon (31, 15, 8) Code in SDR System

Yi Hua Chen, Chang Lueng Chu, Chun Chun Yeh
Oriental Institute of Technology
Institute of Information and Communication Engineering
New Taipei City, Taiwan
ff011@mail.oit.edu.tw, fe029@mail.oit.edu.tw, cindy02175@gmail.com

Abstract—This study used the LabView FPGA to implement the Reed-Solomon codes (R-S code) on the NI SDR PXIe-5641R FPGA module. Besides providing a detailed discussion on the encoding and decoding mechanism of R-S code, this study completed software simulation and hardware verification of R-S (31, 15, 8) code. When the error probability is $10^{-4}$, the coding gain of R-S (31, 15, 8) using $m = 8$ can be up to 2dB. Compared to the R-S (31, 15, 17) code using $m = 5$ [10], when the $E_b/N_0$ is fixed at 5dB, the error probability of is $10^{-2}$; and the error probability in this article is $10^{-3}$, indicating that the R-S (31, 15, 8) implemented in this study has better correction capacity.

Keywords—Reed-Solomon codes; SDR; FPGA

I. INTRODUCTION

Forward error correction (FEC) schemes, different from the source coding for data compression, ensure the correctness of transmission data by restoring the destroyed data. It can be classified into block code and convolution code. Reed-Solomon code is a kind of block code [1] [2]. In December 1958, I. S. Reed and G. Solomon [3] [4] completed in M.I.T Lincoln Laboratory the “polynomial code in the finite field”. The advantages of R-S code include its effective resistance to packet data loss during the network transmission process and its excellent error correction capability. Its disadvantage is the need to use large Galois Field (GF) to establish the long R-S code [5] [6]. Larger number of bits in symbol will result in greater order of power in the information polynomial and higher complexity level of decoding calculation. The algebraic decoding method is commonly used in decoding process [7] [8]. This study used the algebraic BMA algorithm [9] for decoding computation. The error probability was compared with that in Ref. [10], and the LabView simulation was compared with the implementation in LabView FPGA to analyze the effectiveness.

The SDR system used in this study is composed of modules including controller (NI PXIe-8106), transceiver (5641R), down-converter (5600), up-converter (5610). 5641R is an IF transceiver of bandwidth at 20 MHz equipped with DSP optimized Xilinx Virtex-5 SX95T FPGA with IF frequency input and output interface that can be interfaced with analog up and down converter to capture and generate RF signals. The FPGA interface card can be programmed through LabView FPGA to execute complex modulation and signal processing of the hardware.

II. REED-SOLOMON CODES (31, 15, 8) DESIGN PRINCIPLES

A. Reed-Solomon Codes encoding

Reed-Solomon codes are non-binary cyclic codes with symbols made up of m-bit sequences, where m is any positive integer having a value greater than 2.
All the symbols of the m = 8 R-S codes generated by the primitive polynomial \( p(x) = x^8 + x^4 + x^3 + x^2 + 1 \) is shown in Table I.

In general, Reed-Solomon codes can be expressed as R-S \((n, k, t)\) codes, where \( n \) is the total number of code symbols in the encoded block, \( k \) is the number of data symbols being encoded, \( t \) is the symbol-error correcting capability of the code, and \( n - k = 2t \) is the number of parity symbols. The code minimum distance is given by \( d_{\text{min}} = 2t + 1 \). For \( m = 8 \), the codeword length \( n = 2^m - 1 = 255 \), parity symbols \( 2t = n - k = 239 \), code minimum distance, \( d_{\text{min}} = 2t + 1 = 17 \).

**Fig. 1.** R-S code bit architecture SEM

Fig. 1 shows the R-S code structural diagram. The decoding and encoding require 256 GF elements, which can be directly produced by a table circuit. Each codeword block, consisting of source information, \( V(x) \) and protective symbols, is known as the parity check messages, \( P(x) \). The generating polynomial \( g(x) \) produced according to the number of correcting symbol, \( t \), can be represented as \( g(x) = (x + \alpha)(x + \alpha^2) \ldots (x + \alpha^{2t}) \); as \( t = 8 \) in this study, \( g(x) = (x + \alpha^1)(x + \alpha^2) \ldots (x + \alpha^{16}) \). The encoding is to acquire the residual, \( P(x) \), of the long division of \( V(x) \) and \( g(x) \). In order to ensure that the sum of \( V(x) \) and \( P(x) \) can be divided by \( g(x) \) without remainder, the order of \( V(x) \) should be increased. The encoding equation is as shown in (1):

\[
\text{Codewords} = x^{2t} \cdot V(x) + \left\{ V(x) \cdot x^{2t} \right\} \mod g(x) \quad (1)
\]

This paper implements an R-S \((31, 15, 8)\) encoder. The design process is as follows: establish GF\((2^m)\) table (Table I), calculate the coefficients of generating polynomial \( g(x) \), and calculate the modules by dividing \( g(x) \) shown as Fig. 2.

**Fig. 2.** Reed-Solomon Encoder

### B. Reed-Solomon Codes decoding

Some of the codeword will result in error during the transmission process; the received message symbol will be different from the original sending message. In order to identify the locations and values of the error symbols, the process is as follows: syndrome calculator, error location by Chien search algorithm, error values of relevant location using the Forney algorithm. The complete R-S code decoding flowchart is shown in Fig. 3.

![Reed-Solomon Encoder](image)

**Fig. 3.** R-S code decoding process diagram

First, upon receiving the message, Syndrome calculator reveals whether the message contains any error. If syndrome is zero, it means that the message is correct. Syndrome calculator divides the received message by generating polynomial, and it is equivalent to input all the factors of \( g(x) \) including \( \alpha, \alpha^2 \ldots \alpha^{2t} \) into \( r(x) \). If the calculation result is zero, it means that there is no error. The Syndrome calculator is shown in (2):

\[
S_i = r(\alpha^i) = \sum_{j=0}^{2t} r_j (\alpha^j) \quad 1 \leq i \leq 2t \quad (2)
\]

The calculation of error polynomial is the core processing of the entire Reed-Solomon Coding. The BMA algorithm (Berlekamp-Massey Algorithm) uses repeated iterated calculation to calculate the correction polynomial as shown in (3). \( \Lambda^k \) is the correction polynomial, \( k \) is the times of iteration, \( \Delta^i \) is the Delta polynomial.

\[
\sigma(x) = \Lambda^k(x) = \Lambda^{k-1}(x) - \Delta^i T^i(x) \quad (3)
\]
Chien search then is to find out all the locations of the error symbols by putting all \( \alpha^{-i} \) (\( i = 0 \) to 20) into the error location polynomial, \( \sigma(x) \). If the result is zero, it indicates that \( \alpha^{-i} \) is the root of the error location equation \( \sigma(x) = 0 \), the value of \( i \) represents the location of the error symbol of the received message polynomial \( r(x) \). Combining the syndrome polynomial and error location polynomial, we define a key equation, \( \Omega(x) \), as shown in (4).

\[
\Omega(x) = [\sigma(x) \cdot S(x)] \mod X^{2^t} = \sum_{i=0}^{2^t-1} \Omega_i X^{-i} \quad (4)
\]

By Forney equation (4), dividing the first order differentials of \( \sigma(x) \), we can get the error vector \( E(x) \) (5). Finally, according to the error vector \( E(x) \), we can correct the values of the error symbols at the error locations indicated by Chien search method to restore the original accurate message.

\[
E_i = \frac{\Omega_i(\alpha^{-i})}{\alpha^{-i} \cdot \sigma'(\alpha^{-i})} = \frac{\Omega_i(\alpha^{-i})}{\alpha^{-i} \cdot \sigma_{odd}(\alpha^{-i})} \quad (5)
\]

III. IMPLEMENTATION OF REED-SOLOMON CODES (31, 15, 8)

The encoding part establishes the primitive polynomial table to set the parameters. R-S decoding program contains sub-programs of the four steps of syndrome, error location, Chien search and Forney algorithm. The error location uses the BMA algorithm and Chien search are the most complex. In programming the calculation process, the script file should be broken into a few sub-programs separately for the application in BMA algorithm. Chien search works along with the final Forney algorithm program. As it may take up large amount of hard disc space, the Boolean elements are generated by programs to comply with its characteristics.

The sub-programs are completed and summarized as shown in Fig. 4. Coupled with some judgment and control pins, the entire decoding can be executed. During the decoding process, after executing a block, the program needs to be reset to enter into next block, thus, the program needs to make automatic judgment. The decoding sequence is syndrome value, BMA, Chien search and Forney algorithm.

IV. SOFTWARE VERIFICATION OF REED-SOLOMON CODES

The error performance of R-S codes of (31, 15, 8) with BPSK modulation in AWGN environment is shown in Fig. 5. When the error probability is \( 10^{-4} \), the R-S (31, 15, 8) coding gain can be up to 2dB.

Regarding the R-S (31, 15, 17) codes used by Lin, 17 is \( d_{min} = 2t + 1 \), therefore, \( t = 8 \) and the R-S (31, 15, 8) in this study are the same codes. However, \( m = 5 \), it indicating that a symbol contains that is consisted of 5 bits and can correct 40 error bits; while can be corrected. The method proposed in this study can correct 64 error bits. If the \( E_b/N_0 \) is kept at 5 dB in the same AWGN channel environment, the error probability of R-S (31, 15, 8) is about \( 10^{-3} \) (blue solid square), and the error probability of R-S (31, 15, 17) is about \( 10^{-2} \) (the green circle) [10], it means the error probability of R-S (31, 15, 8) is lower than that of R-S (31, 15, 17) by about \( 10^{-1} \) at the same transmission power.

![Figure 5. Reed-Solomon Codes (31, 15, 8) error probability performance analysis.](image)

V. CONCLUSIONS

![Figure 6. Reed-Solomon Code (31, 15, 8) decoder compilation status](image)
The LabView FPGA compilation status of Reed-Solomon code (31, 15, 8) of this study is shown in Fig. 6. The program performance is compared with the Reed-Solomon OTN (Optical Transport Network) Decoder [11] shown in Fig. 7.

<table>
<thead>
<tr>
<th>Blocks</th>
<th>Total Available</th>
<th>32 RS</th>
<th>16 RS</th>
</tr>
</thead>
<tbody>
<tr>
<td>LUTs</td>
<td>44,800</td>
<td>18,383</td>
<td>12,270</td>
</tr>
<tr>
<td>Registers</td>
<td>44,800</td>
<td>35,144</td>
<td>23,991</td>
</tr>
<tr>
<td>Block Memory</td>
<td>148</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>Response Time</td>
<td>-</td>
<td>5 ms</td>
<td>5.1 ms</td>
</tr>
</tbody>
</table>

Figure 7. Reed-Solomon OTN decoder compilation status

16 RS represents the decoding hardware architecture contained sixteen BMA, sixteen error locator and sixteen error evaluator blocks. These blocks are usually to occupy a lot of hardware space. Compared with 32 RS decoding hardware architecture, LUTs (Look-Up Table) is 14% reduced (for 16 RS 27.3% and 32 RS 41.0%), Registers is 25% down (for 16 RS 53.3% and 32 RS 78.4%). Nevertheless, the LUTs is 1.5% and Registers is 1.3% for R-S (31, 15, 8) studied in this paper. But in processing clock rate, Reed-Solomon OTN Decoder [11] is 166 MHz, which is a little bit higher than the 161 MHz of R-S (31, 15, 8) code.

REFERENCES

ICCSNT 2012
2nd Int. Conference on Computer Science and Network Technology

ICCSNT2011  About Changchun  About NENU

Sat-Mon, Dec 29-31, 2012
Northeast Normal University, CHANGCHUN, CHINA

Latest News

- [2012-08-01] ICCSNT 2012 has been listed in China Science Meeting Online. [click]
- [2012-07-20] IEEE Publication Information

<table>
<thead>
<tr>
<th>IEEE Catalog Number:</th>
<th>CFP1209R-CDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISBN:</td>
<td>978-1-4673-2962-0</td>
</tr>
</tbody>
</table>

- [2012-07-19] ICCSNT2012 has been listed in IEEE Conference Search. [click]
- [2012-06-25] Website of ICCSNT 2012 has been launched.

About

ICCSNT2012 will be held on Dec 29, 2012. It is the second forum for the presentation of new advances and research results in a wide variety of scientific areas with a common interest in improving Future Computer Science, Network Technology and Communication related techniques. The conference is a annual conference series, focusing on both theory and applications mainly covering the topic of Computer Science, Network Technology, Information Technology, Communication. In addition to the technical sessions, there will be invited sessions, tutorials and keynote addresses.

ICCSNT2012 is sponsored by Northeast Normal University. The objective of this Conference is to offer a meeting point for professionals and researchers from industry and academic to present and discuss recent advances in the field.


2013/2/20
Utility-Based Load Distribution for QoS Provisioning and Utility Maximization in Wireless Random Access Networks

Bo Gu*, Cheng Zhang*, Kyoko Yamori†, Sugang Xu*, Yoshinori Tanaka‡

*Global Information and Telecommunication Institute, Waseda University, Tokyo, 169-0051 Japan
†Department of Information Management, Asahi University, Gifu, 501-0223 Japan
‡Research Institute of Science and Engineering, Waseda University, Tokyo, 162-0044 Japan
Email: {bo.gu, cheng.zhang}@akane.waseda.jp, kyamori@alice.asahi-u.ac.jp, xusugang@m.teicc.org, ytanaka@waseda.jp

Abstract—In this paper, a utility-based load distribution approach is introduced for relieving congestion at certain popular spaces within the network. Congestion-indication signals are provided for suggesting users to select APs in response to network load conditions. The heavier traffic load an AP has, the lower utility each associated user gains. Hence, the users incentivize themselves to associate to less congested APs. The effectiveness of the algorithm on improving the degree of load balance is evaluated using simulations. Simulation results show that the proposed algorithm achieves greater balance and higher resource utilization when compared with the best existing algorithms.

I. INTRODUCTION

In dense metropolitan areas, it is often the case that a user can detect several APs simultaneously. However, users tend to associate themselves to the AP that has the strongest Received Signal Strength Indicator (RSSI). A key consequence of these behaviors is that the traffic load is often distributed unevenly among the APs [1], [2], [3], [4].

To address this problem, a utility-based load distribution approach is introduced and analyzed. It is assumed that APs deployed by different businesses are able to select to join into a federated network. A Central Aggregation Server (CAS) is deployed in the federated network to maintain network status by periodically collecting the load level at each AP. The CAS receives the request from each user and helps to find APs that can accommodate the service request [5].

To be specific, the mission of CAS is to help identifying two candidate APs (if exist) for each user: (i) a candidate AP whose service could be provided in place (hereafter termed local candidate); and (ii) a candidate AP somewhere else whose service could be provided for the user while requiring user's physical roaming (hereafter termed remote candidate).

The proposed algorithm trades off signal strength with load by suggesting a user to change the association from an overloaded AP with stronger signal (i) to a lightly loaded local candidate with possibly weaker signal; or (ii) to a lightly loaded remote candidate whose service is provided requiring user's physical roaming.

The heavier traffic load the AP has, the lower utility each associated user gains. Hence, the users incentivize themselves to associate to less congested APs.

II. RELATED WORK

The studies [6], [7], [8], [9] focus on using per-flow resource-based admission control combined with prioritized data transmission for real-time traffic, thus improving user's QoS within a single cell in the network. The main drawback of these approaches is that the dynamics of traffic load, in terms of both time of a day and location [10], are not sufficiently considered. Hence, that the traffic load still tends to be distributed unevenly among the APs and the QoS cannot be guaranteed very well at heavily loaded APs.

As studied in work [11], [12], [13], the load imbalance problem can be alleviated by balancing the load among the APs via intelligently selecting the user-AP association. The authors suggest adding load information, such as the packet loss rate, the throughput, the retransmission probability or the airtime cost [13], to the beacon frames. Users should use this information in addition to the signal strength to select the APs. Although we agree that these approaches show nice features, we think that there are two drawbacks to overcome: (i) these approaches distribute users only across available overlapping cells, hence that the load could be only locally balanced; and (ii) due to the lack of inter-operability between APs deployed by different businesses, the APs do not cooperate to balance the load across the network.

In [14], authors focus on distributing traffic load within the network using a centralized approach. A centralized server is deployed to monitor the bandwidth allocation in the entire network. The centralized server then helps to identify an AP where an incoming user's QoS bound (i.e., a minimum and
Infection Analysis Using Colour Feature Texture Using Image Processing

Ravikant Sinha\textsuperscript{1}
Department of E&T C Pune
University, SCOEO sadumbare
Maharashtra, India

Sandhya Sinha\textsuperscript{2}
Department of E&T C Pune
University, RMDSSOE Warje
Maharashtra, India

Abstract—In this study, a new approach is used to automatically detect the infected pomegranates. In the development of automatic grading and sorting system for pomegranate, critical part is detection of infection. Color texture feature analysis is used for detection of surface defects on pomegranates. Acquired image is initially cropped and then transformed into HSI color space, which is further used for generating SGDM matrix. Total 18 texture features were computed for hue (H), saturation (S) and intensity (I) images from each cropped samples. Best features were used as an input to Support Vector Machine (SVM) classifier and tests were performed to identify best classification model. Out of selected texture features, features showing optimal results were cluster shade (99.8835%), product moment (99.8835%) and mean intensity (99.8059%).

Keywords—Pomegranate, disease detection, machine vision, color co-occurrence method, SGDM, texture features.

I. INTRODUCTION

The pomegranate (Punica granatum) is a fruit-bearing deciduous shrub or small tree that grows to between five and eight metres tall and is best suited to climates where winters are cool and summers are hot. The pomegranate is thought to have been first cultivated 5 to 6,000 years ago and is native to the regions from Iran through to north India. It is now widely cultivated throughout Eastern Europe, Asia and the USA, the main areas of world production being in India, Iran, Spain and California. Pomegranates can be consumed as fresh fruit or used in fruit juices, teas, pharmaceutical and medicinal products and in dyes or as decoration. There are several different varieties of pomegranate recognised in Iran alone and even more globally, some of the cultivars that have the greatest impact are ‘Molber’, ‘Ahmar’, ‘Bhagawa’, ‘Hicazari’ and ‘Dente di cavallo’. Globally, it is estimated that total production amounts to around 2,000,000 tonnes, of which India produces approximately 50% in the states of Maharashtra and Andhra Pradesh. Iran is the second largest, producing around 35% of global production. Spain produces around 2.5% and the USA has around 10,000ha under production. This study centres at developing a method to detect infected pomegranate using color texture features. The various steps involved for development of database of features are summarized in figure 1 below:

![Diagram of process](Image)

II. LITERATURE REVIEW

Thomas J. Burks et al (2009) [1] demonstrated that color imaging and texture feature analysis could be used for classifying citrus peel diseases under the controlled laboratory lighting conditions. The present work is an extension of that research, providing a feasibility analysis of the technology in classification of infected pomegranates. Edwards and Sweet (1985) [2] used reflectance spectra of the entire citrus plant for estimating the damage caused due to citrus blight. Hetal Patel et al (2011) [3] designed the algorithm aiming at calculation of different weights for features like intensity, color, orientation and edge of the test image. S.Arivazhagan et al (2010) [4] used computer vision strategy to recognize a fruit...
ICCSNT 2012 Organizing Committee

Honorary Chair
Bodo Urban PhG- IGD, Germany

General Chair
Shaochun ZHONG Northeast Normal University, China

Organizing Committee Chair
Dongdai ZHOU Northeast Normal University, China

Organizing Committee Co-Chairs
Shuli SUN Heilongjiang University
Hongji YANG De Montfort University, United Kingdom
Xiaochun CHENG Middlesex University, United Kingdom
Yueguang XIE Northeast Normal University, China
Ming HUANG Dalian Jiaotong University, China
Xiaodan LIU Liaoning Normal University, China
Dimitrios A. Karras Chalkis Institute of Technology, Hellas

Technical Committee Chairs
Wenyong WANG Northeast Normal University, China
Chengzhong XU Wayne State University, United States
Kun YANG University of Essex, United Kingdom

Technical Committee
Wei PAN Northeast Normal University, China
Xiaojuan ZHENG Northeast Normal University, China
Zhiyi FANG Northeast Normal University, China
Jianchao XU Northeast Normal University, China
Guohui YANG Harbin Institute of Technology, China
Haiying WANG Harbin University of Science and Technology, China
Mingfeng WU University of Texas Austin, USA
Lukui JIN  
Mostafa Sadeghi  
Marilza Antunes de Lemos  
Faezeh Memarzadeh-Zavareh  
Usha Subburaj  
Supoj Hengpraoprohm  
Ali Mahdeian  
Mostafa Abdollahian-Dehkordi  
Mohammad Memarzadeh-Zavareh  
Fusuhiro Matsuda  
Ahad Jahangiry  
Mohammad Sabokrou  
Shahram Babaie  
Amjad Mehmood  
Rabia Khan  
El'mira Kalimulina  
Hyung-Woo Lee  
Sadaqat Rehman  
Arifur Rahman  
Krishna Baishnab  
Satheesh Suthani  
Iman Saberi  
Akash Singh  

**Finance and Registration Chair**

Chen LIU  
Northeast Normal University, China
## Author Index—Volume 1

### A

<table>
<thead>
<tr>
<th>Page</th>
<th>Name</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>53</td>
<td>Ai Ting</td>
<td>0093</td>
</tr>
<tr>
<td>555</td>
<td>Ai Ting</td>
<td>2630</td>
</tr>
<tr>
<td>566</td>
<td>Ai Ting</td>
<td>2658</td>
</tr>
<tr>
<td>656</td>
<td>Ai Ting</td>
<td>2842</td>
</tr>
<tr>
<td>275</td>
<td>Aihua Yin</td>
<td>2247</td>
</tr>
<tr>
<td>25</td>
<td>Alma-Ata, Kazakhstan</td>
<td>0056</td>
</tr>
<tr>
<td>708</td>
<td>An Wang</td>
<td>0115</td>
</tr>
<tr>
<td>694</td>
<td>Angli Liu</td>
<td>0024</td>
</tr>
</tbody>
</table>

### B

<table>
<thead>
<tr>
<th>Page</th>
<th>Name</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Bektemysova Gulnar</td>
<td>0056</td>
</tr>
<tr>
<td>433</td>
<td>Bin Han</td>
<td>2455</td>
</tr>
<tr>
<td>321</td>
<td>Binbin Yu</td>
<td>2307</td>
</tr>
<tr>
<td>406</td>
<td>Bo Gu</td>
<td>2431</td>
</tr>
<tr>
<td>386</td>
<td>Bo Huang</td>
<td>2422</td>
</tr>
</tbody>
</table>

### C

<table>
<thead>
<tr>
<th>Page</th>
<th>Name</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CAI Lei</td>
<td>0013</td>
</tr>
<tr>
<td>598</td>
<td>Cai Wu Wang</td>
<td>2682</td>
</tr>
<tr>
<td>158</td>
<td>Cao Liying</td>
<td>2125</td>
</tr>
<tr>
<td>681</td>
<td>Cao Qinghua</td>
<td>0012</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Page</th>
<th>Name</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>367</td>
<td>Cao Xinyan</td>
<td>2370</td>
</tr>
<tr>
<td>198</td>
<td>Cao Xin-Yan</td>
<td>2189</td>
</tr>
<tr>
<td>465</td>
<td>Chang Lueng Chu</td>
<td>2530</td>
</tr>
<tr>
<td>660</td>
<td>Changju Kan</td>
<td>2863</td>
</tr>
<tr>
<td>193</td>
<td>Chang-xiong Xia</td>
<td>2181</td>
</tr>
<tr>
<td>266</td>
<td>Chao JIANG</td>
<td>2231</td>
</tr>
<tr>
<td>579</td>
<td>Chao Xing</td>
<td>2673</td>
</tr>
<tr>
<td>602</td>
<td>Chao Xing</td>
<td>2683</td>
</tr>
<tr>
<td>45</td>
<td>Chen Jian</td>
<td>0084</td>
</tr>
<tr>
<td>639</td>
<td>Chen Lingyi</td>
<td>2731</td>
</tr>
<tr>
<td>162</td>
<td>Chen Qinghua</td>
<td>2130</td>
</tr>
<tr>
<td>452</td>
<td>CHEN Xiu-liang</td>
<td>2493</td>
</tr>
<tr>
<td>666</td>
<td>Chen Yisong</td>
<td>2871</td>
</tr>
<tr>
<td>671</td>
<td>Chen Yisong</td>
<td>2872</td>
</tr>
<tr>
<td>652</td>
<td>CHENG Dao-wen</td>
<td>2840</td>
</tr>
<tr>
<td>656</td>
<td>CHENG Dao-wen</td>
<td>2842</td>
</tr>
<tr>
<td>225</td>
<td>Cheng Yuxi</td>
<td>2210</td>
</tr>
<tr>
<td>406</td>
<td>Cheng Zhang</td>
<td>2431</td>
</tr>
<tr>
<td>336</td>
<td>Chengxi XU</td>
<td>2331</td>
</tr>
<tr>
<td>411</td>
<td>Ching-Yang Lin</td>
<td>2432</td>
</tr>
<tr>
<td>391</td>
<td>Chong Poh Kit</td>
<td>2426</td>
</tr>
</tbody>
</table>