ՀՏԴ 004.655:621.396.933

Ռ.Կ. ԱՊԻԿՅԱՆ

ԱՐԲԱՆՅԱԿԱՅԻՆ ՌԱԴԻՈՆԱՎԻԳԱՑԻՈՆ ՀԱՄԱԿԱՐԳԻ՝ C/A ԿՈԴՈՎ ՆԱՎԻԳԱՑԻՈՆ ՏՎՅԱԼՆԵՐԻ ՄՈԴՈՒԼԱՑՄԱՆ ԵՎ ԱՊԱՄՈԴՈՒԼԱՑՄԱՆ ՍԵԳՄԵՆՏԻ ՄՇԱԿՈՒՄՆ ՈՒ ՄՈԴԵԼԱՎՈՐՈՒՄԸ JAVA ՄԻՋԱՎԱՅՐՈՒՄ

Ներկայացվում է արբանյակային ռադիոնավիգացիոն համակարգի C/A կոդի և տեքստային տվյալների մոդուլացման և ապամոդուլացման գործընթացը, ինչպես նաև հաղորդման և ընդունման համար անհրաժեշտ գեներատորների, մոդուլարարների, ապամոդուլարարների, դետեկտորների, տեքստայինից երկուական և երկուականից տեքստային տվյալների կերպափոխիչների մոդելավորումը, կառուցումը JAVA ծրագրային լեզվով Intellij IDEA միջավայրում:

Առանցքային բառեր. GPS, Java, C/A կոդ, «ոսկե» համար, C/A կոդի գեներատոր, C/A ընդունիչ, WiFi, մոդուլացում, ապամոդուլացում:

Ներածություն: GPS համակարգում հանրությանը հասանելի նավիգացիոն տվյալները կոդավորվում են C/A կոդով և հաղորդվում L1 կրող ազդանշանի միջոցով: C/A կոդը բացառիկ է ցանկացած արբանյակի համար և գեներացվում է որոշակի ալգորիթմով, որում որպես մուտքային տվյալ է հանդիսանում արբանյակի համարը: Ալգորիթմի իրականացումից հետո տվյալները մոդուլավորվում են ստացված C/A կոդով: Արդեն կոդավորված տվյալները հաղորդվում են հաջորդ մոդուլ, որտեղ տեղի է ունենում BPSK մոդուլացում L1 կրողով և կոդավորված տվյալներով, որից հետո ստացված ազդանշանը հաղորդվում է անտենային: Ընդունիչը կատարում է ստացված ազդանշանի ապակոդավորում և հայտնաբերում է արբանյակի համարը: Հաջորդիվ իրականցվում է C/A կոդով կոդավորված տվյալների ապակոդավորում, և ստացվում է նավիգացիոն տվյալների երկուական տեսքը [1]:

Աշխատանքում վերլուծվում են վերը նկարագրված փուլերը և C/A կոդի գեներացումը, որին զուգահեռ մշակվում են երկու ծրագիր Java միջավայրում, որոնցից առաջինը կլինի հաղորդող մասը, որը թույլ կտա գեներացնել C/A կոդը, համապատասխան ընտրված արբանյակի համարի, կոդավորել մուտքագրված տվյալները ստացված C/A կոդով և հաղորդել ստացված արդեն կոդավորված տվյալները WIFI ցանցով: Վերջնական տվյալները կուղարկվեն TCP տրանսպորտային փաթեթի միջոցով: Երկրորդը կլինի ընդունող մասը, որը կընդունի կոդավորված

տվյալները և կպատկերի հաղորդած արբանյակի համարը և նավիգացիոն տվյալները:

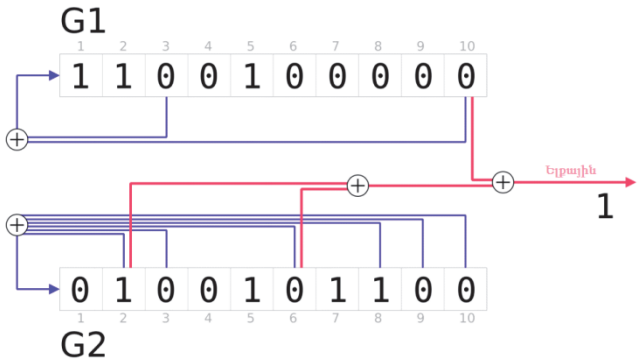
Խնդրի դրվածքը և մեթոդական հիմնավորումը: C/A կոդի կառուցվածքը և գեներացման ալգորիթմը: C/A կոդը ներկայացնում 0-ների և 1-երի քվազի պատահական բիթերի հաջորդականություն: C/A կոդի բիթերը իրականում տեղեկույթ չեն պարունակում: Այս փաստից ելնելով՝ դրանք կոչվում են նաև չիպեր: Ստացված կոդի պարունակությունը կախված է արբանյակի համարին համապատասխան ընտրված «ոսկե» թվերի կոմբինացիայից, որոնք բերված են աղ. 1–ում [1,2]:

Աղյուսակ 1

«Ոսկե» թվերը՝ արբանյակի համարին համապատասխան

Արբանյակի համարը	Համապատասխան բիթերի ելքային կոմբինացիան	Արբանյակի համարը	Համապատասխան բիթերի ելքային կոմբինացիան
1	2 ⊕ 6	17	1 ⊕ 4
2	3 ⊕ 7	18	2 ⊕ 5
3	4 ⊕ 8	19	3 ⊕ 6
4	5 ⊕ 9	20	4 ⊕ 7
5	1 ⊕ 9	21	5 ⊕ 8
6	2 ⊕ 10	22	6 ⊕ 9
7	1 ⊕ 8	23	1 ⊕ 3
8	2 ⊕ 9	24	4 ⊕ 6
9	3 ⊕ 10	25	5 ⊕ 7
10	2 ⊕ 3	26	6 ⊕ 8
11	3 ⊕ 4	27	7 ⊕ 9
12	5 ⊕ 6	28	8 ⊕ 10
13	6 ⊕ 7	29	1 ⊕ 6
14	7 ⊕ 8	30	2 ⊕ 7
15	8 ⊕ 9	31	3 ⊕ 8
16	9 ⊕ 10	32	4 ⊕ 9

C/A կոդի գեներացման հիմքում ընկած է երկու տեղափոխման ռեգիստրների աշխատանքը (G1, G2), որոնցից յուրաքանչյուրի տարողությունը 10 բիթ է: G1 ռեգիստրի աշխատանքը կախված չէ մուտքային տվյալից, այսինքն՝ արբանյակի համարից: Այն միշտ գեներացնում է միևնույն կոդը: G2 ռեգիստրի աշխատանքը և ելքային տվյալները, ի տարբերություն G1 ռեգիստրի, կախված են արբանյակի համարից: Ռեգիստրների աշխատանքի ընդհանուր ալգորիթմը պատկերված է նկ 1 –ում:



Նկ. 1. G1 և G2 ռեգիստրների կոդի գեներացման ալգորիթմի ընդհանուր սխեման

Գեներացումից առաջ երկու ռեգիստրների բիթերի արժեքները հավասարեցվում են մեկի:

G1 ռեգիստրի ելքային բիթերի ստացման ալգորիթմը: G1 ռեգիստրի ելքային բիթերի գեներացիան կատարվում է հետևյալ ալգորիթմով՝

$$1 + x^3 + x^{10},$$

որտեղ X^3 և X^{10}-ը ռեգիստրի համապատասխան 3-րդ և տասներորդ բիթերն են, որոնց արժեքները ենթարկվում են բացառիչ ԿԱՄ-ի, իսկ ստացված արդյունքը տեղադրվում է ռեգիստրում որպես առաջին բիթ: Մնացած բիթերը մեկ սեգմենտով տեղափոխվում են աջ: Յուրաքանչյուր նման քայլում աջ տեղաշարժից հետո 10-րդ բիթը դուրս է գալիս որպես ելքային տվյալ [2]:

G2 ռեգիստրի ելքային բիթերի ստացման ալգորիթմը: G2 ռեգիստրի կոդի գեներացման ալգորիթմը ներկայացվում է հետևյալ արտահայտությամբ՝

$$1 + x^2 + x^3 + x^6 + x^8 + x^9 + x^{10}:$$

G1 ռեգիստրի նման՝ այս արտահայտությունը ցույց է տալիս այն բիթերի համարները, որոնք ենթարկվում են բացառիչ ԿԱՄ գործողությանը: Արդյունքը գալիս է ռեգիստրի առաջին բիթ, մնացած բիթերը ենթարկվում են մեկական աջ տեղաշարժի, սակայն այս դեպքում որպես ելքային արժեք վերցնում ենք ոչ թե վերջին բիթը, այլ որպես մուտքային պարամետր նախօրոք տրված ոսկե համարին համապատասխան երկու բիթերի բացառիչ ԿԱՄ-ի արժեքը: Նկ. 1 –ում պատկերված է G2 ռեգիստրի ելքային բիթերի կոմբինացիան 2 և 6 բիթերի դեպքում: 2-ը և 6-ը առաջին արբանյակի «ոսկե» համարներն են՝ ըստ աղ. 1 – ի [2]:

C/A կոդի գեներատոր: C/A կոդի գեներատոր լինելու է ինչպես ընդունող, այնպես էլ հաղորդող մասի ծրագրային փաթեթներում: Գեներատորը պարունակում է երկու ռեգիստր (G1, G2): Ծրագրային փաթեթում այս ռեգիստրները կոչվում են “Polinomial1” (G1) և “Polinomial2” (G2): Երկուսն էլ հիմնված են տեղափոխման ռեգիստրի աշխատանքի վրա և տարբերվում են միայն մուտքային ալգորիթմի տեսակով (օրինակ՝ $1 + x^3 + x^{10}$ կամ $1 + x^2 + x^3 + x^6 + x^8 + x^9 + x^{10}$) և ելքային բիթի ստացման մեխանիզմով: Այս նմանությունները հաշվի առնելով՝ ստեղծում ենք “PolinomialProcessor.java” դասը, որը պարունակում է տասը բիթանի տեղափոխման ռեգիստր, որը կարող է աշխատել տրված X ալգորիթմով և ունի աբստրակտ մաս ելքային արժեքի համար: “Polinomial1” (G1) և “Polinomial2” (G2) գեներատորները ժառանգում են “PolinomialProcessor.java” դասը [3], և ամեն մեկը ներմուծում է իր ելքային պարամետրի ստացման տրամաբանությունը (այդ դասերը կարելի է գտնել “polinomial” թղթապանակ մեջ) [4]:

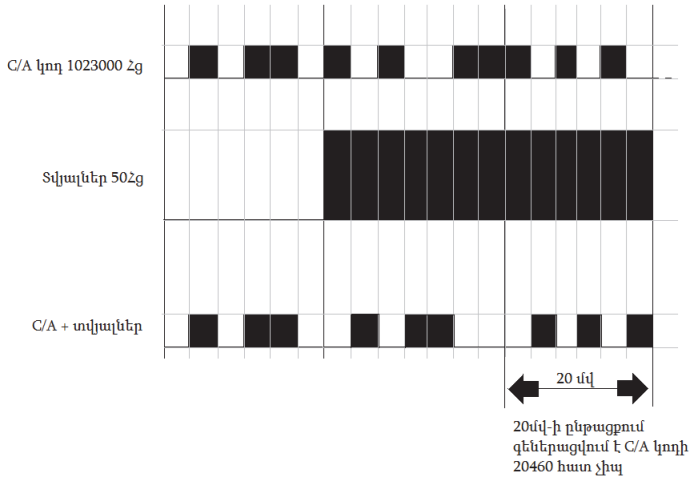
Ստացված C/A կոդի և նավիգացիոն տվյալների մոդուլավորումը: GPS համակարգում տվյալները հաղորդվում են 50*Հց* հաճախությամբ, որը 50 *բիթ/վ* արագությունն է: Այս արագությունը շատ ավելի փոքր է, քան C/A կոդի բիթային արագությունը, որը 1023000 *Հց* է կամ 1023000 *չիպ/վ*: Ասվածից հետևում է, որ մոդուլացումից հետո հաղորդագրության մեկ բիթը պարունակելու է 20460 չիպ C/A կոդ, որը երևում է նկ. 2 -ում:

Տվյալները և C/A կոդը մոդուլվում են աղ. 2-ում բերված օրինաչափությամբ:

Աղյուսակ 2

Տվյալների և C/A կոդի մոդուլման օրենքը

Տվյալ	C/A	արդյունք
1	0	1
1	1	0
0	1	1
0	0	0



Նկ. 2. Տվյալների մոդուլումը կամ կոդավորումը C/A կոդով

C/A մոդուլարարի ծրագրային մոդելավորումը: C/A մոդուլարարը պետք է պարունակի C/A կոդի գեներատոր, իսկ նրա մուտքին տրվում են արբանյակի համարին համապատասխան «ոսկե» համարը և տվյալների վերաբերյալ տեղեկատվությունը երկուական տեսքով: Արդեն նշվել է, որ տվյալների մեկ բիթը պարունակում է 20460 հատ C/A կոդի չիպ: Մոդուլացման սկզբում C/A մոդուլարարը գեներացնում է 20460 հատ C/A կոդի չիպ և մոդուլում այն տվյալների առաջին բիթի հետ աղ. 2 –ում բերված օրենքով: Այնուհետև C/A կոդի գեներատորը բերվում է սկզբնական տեսքին, այսինքն՝ ռեգիստրների բոլոր բիթերի արժեքները հավասարվում են 1 –ի: Այնուհետև կատարվում է նույն գործողությունը տվյալների հաջորդ բիթի համար: Արդյունքում ստացվում է C/A կոդով մոդուլված ազդանշան, որը տրվում է հաղորդող դասի մուտքին: C/A կոդի մոդուլարարի դասի նկարագրությունը կարելի է գտնել “modulators” թղթապանակի մեջ [4]:

Ընդունող մասի կառուցվածքը: Ընդունող մասը պարունակում է C/A կոդի դետեկտոր, C/A կոդի ապամոդուլարար և երկուական տվյալներից տեքստայինին անցման կերպափոխիչ [5]:

C/A կոդի դետեկտոր: Դետեկտորի խնդիրն է հայտնաբերել ստացված C/A կոդով մոդուլված տվյալների միջից արբանյակի համարին համապատասխանող «ոսկե» համարը [5]:

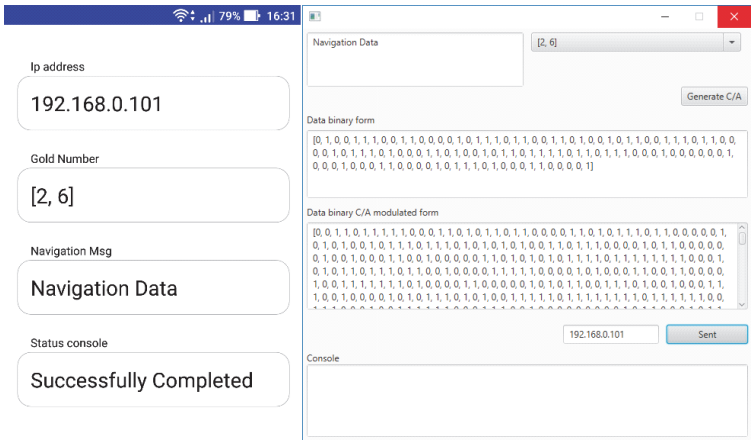
Այլ կերպ ասած, հայտնաբերել, թե որ արբանյակից է ուղարկվել ազդանշանը: Դետեկտորը պարունակում է C/A կոդի գեներատոր: Դետեկտումը կա-

տարվում է փուլերով: Յուրաքանչյուր փուլում կատարվում է հերթական արբանյակի C/A կոդի գեներացում 20460 չիպի համար, որից հետո կատարվում է համեմատում ստացված C/A կոդի և ստացած կոդավորված տվյալների միջև: Համեմատելու ընթացքում գրանցվում է, թե քանի անգամ է հայտնաբերվել գեներացված C/A կոդը կոդավորված տվյալների մեջ: Այս պրոցեսը կատարվում է յուրաքանչյուր արբանյակի դեպքում: Եթե կազմենք աղյուսակ արբանյակի C/A կոդի և նրան համապատասխան համընկնումների համար, ապա մաքսիմում համընկնումների քանակ հավաքած C/A կոդը կլինի հաղորդող արբանյակի C/A կոդը: Համապատասխան C/A կոդի՝ կարող ենք վերցնել արբանյակի համարին համապատասխան «ոսկե» համարը (այս դասերը կարող եք գտնել “detectors” թղթապանակի մեջ):

C/A կոդի ապամոդուլյատոր: Ապամոդուլարարը որպես մուտքային արժեք է ընդունում դետեկտորի միջոցով հայտնաբերված C/A կոդը: Դետեկտման փուլում C/A կոդը համեմատվում է կոդավորված տվյալների առաջին 20460 բիթերի հետ: Եթե համընկնում է, ապա ելքային արժեքը 0 – է, իսկ եթե համընկնում չկա, գեներացնում է նույն C/A կոդի հակառակ կամ “reverse” տարբերակը (C/A = 01001001… C/A reverse = 10110110…) և համեմատվում է, եթե համընկնում կա, ապա ելքային բիթը հավասար կլինի 1-ի: Այս գործողությունը կատարվում է տվյալների յուրաքանչյուր 20460 բիթի համար: Արդյունքում՝ ստացվում է ապակոդավորված տվյալների երկուական տեսքը, որը կերպափոխում ենք տեքստայինի և պատկերում էկրանին (այս դասերը կարող եք գտնել “demodulators” թղթապանակի մեջ) [5]:

Ծրագրի փորձարկումը: Մինչև ծրագրերի փորձարկումը՝ ներբեռնենք դրանք [4] և [5] համապատասխան հասցեներից: Համակարգչի վրա բացենք “C/A generator” ծրագիրը [4]: Ընդունող կողմը կլինի Android համակարգով աշխատող սարքը, որի վրա պետք է բացել “GPS CA receiver” ծրագիրը (նկ.3) [5]: Համակարգչի վրա բացված “C/A generator” (նկ.3-ի ա.) ծրագրի մուտքային դաշտում ներմուծենք որոշակի տեքստ: Փորձի շրջանակներում այս տեքստը հանդես է գալիս որպես նավիգացիոն հաղորդակցություն: Նշենք արբանյակի համարին համապատասխան «ոսկե» համարը և սեղմենք ”Generate C/A” կոճակը: “Data binary form” և “Data binary C/A modulated form” դաշտերում գեներացվում և պատկերվում են համապատասխանաբար մուտքագրված տեքստի երկուական տեսքը և C/A կոդով մոդուլացված տեսքը: Հաջորդիվ՝ ընդունող կողմում բացենք “C/A receiver” ծրագիրը (նկ. 3 -ի բ.): “C/A generator” ծրագրի մեջ՝

“IP address” դաշտում, ներմուծենք “C/A receiver” ծրագրի մեջ պատկերված հասցեն: Սեղմենք “C/A generator” ծրագրի “Send” կոճակը: Կտեսնենք, որ սկսվում է տվյալների հաղորդումը դեպի ընդունիչ կողմ: Ընդունիչ կողմում ցույց է տրվում ներկա պահին դետեկտման գործընթացը, և պատկերվում է համընկնումների քանակը, այսինքն՝ հերթական C/A կոդի համար քանի համընկնում կա: Որոշակի ժամանակ հետո ուղարկված տեքստային հաղորդագրությունը պատկերվում է էկրանին:



ա. *բ.*

Նկ. 3. Հաղորդվող և ընդունող մասերի ծրագրային պատուհանները.

ա. “C/A generator”, բ.”GPS C/A receiver”

Եզրակացություն: Java միջավայրում մոդելավորվել և գրվել է ծրագիր, որը թույլ կտա ներմուծված տեքստային տվյալները վերածել երկուականի, մոդուլավորել արբանյակի համարին համապատասխան C/A կոդով և հաղորդել այն WIFI ցանցով դեպի ընդունիչ: Ընդունման կետում ընդունած տվյալներից դետեկտվում է արբանյակի համարը, ապամոդուլացվում, վերածվում տեքստային UTF-8 ֆորմատի և պատկերվում է հաղորդագրությունը էկրանին:

ԳՐԱԿԱՆՈՒԹՅԱՆ ՑԱՆԿ

1. James Boa-Yen Tsui, GPS C/A Code Signal Structure, հատոր համար 5 (link: http://read.pudn.com/downloads85/ebook/326017/Fundamentals%20of%20Global%20Positioning%20System%20Receivers/booktext05.pdf)
2. Electrolyte (YouTube channel https://www.youtube.com/watch?v=cxr3youvQ9g)
3. JAVA documentation
4. (https://docs.oracle.com/javase/7/docs/api/)

5. “GpsGenerator” ծրագրի հղումը: https://github.com/RobertApikyan/ HYPERLINK "https://github.com/RobertApikyan/GpsGenerator"GpsGenerator
6. “GPS C/A receiver” ծրագրի հղումը: https://github.com/RobertApikyan/GPS-C-A-receiver

Р.К. АПИКЯН

РАЗРАБОТКА И МОДЕЛИРОВАНИЕ СЕГМЕНТА МОДУЛЯТОРА И ДЕМОДУЛЯТОРА С/А КОДА И ТЕКСТОВЫХ ДАННЫХ В СРЕДЕ JAVA СПУТНИКОВОЙ РАДИОНАВИГАЦИОННОЙ СИСТЕМЫ

Рассматриваются процессы модуляции и демодуляции текстовых данных с помощью С/А кода спутниковых радионавигационных систем, а также моделирования и построения данных необходимых для передачи и приема на основе программы Java в среде IntellijIDEA.

Ключевые слова: GPS, Java, С/А код, “золотые” цифры, генератор С/А кода, С/А приемник**,** WiFi**,** модуляция, демодуляция.

R.K. APIKYAN

NAVIGATION SYSTEM C/A CODE AND NAVIGATION DATA MODULATION AND DEMODULATION SEGMENTS CONSTRUCTION WITH JAVA

The GPS C/A code and navigation data modulation and demodulation process is presented. The program is written with Java IntelliJ IDEA environment.

Keywords: GPS, Java, C/A code, Gold numbers, C/A code generator, C/A receiver**,** WiFi**,** modulation, demodulation.