



รายงานวิจัยฉบับสมบูรณ์

การใช้คอมพิวเตอร์ เพื่อศึกษาความเป็นไปได้ของการเกิดปรากฏการณ์ควอนตัม
คอนไฟน์เมนต์ในหลอดควอนตัมของแหล่งกำเนิดแสงชนิดสารกึ่งตัวนำ
A computer simulation search for the quantum confined Stark effect in a
semiconductor light source with the quantum well.

โดย ดร.กัทร อัยรักษ์ มหาวิทยาลัยสงขลานครินทร์
รศ.บุญเหลือ พงศ์ดารา มหาวิทยาลัยสงขลานครินทร์ (นักวิจัยพี่เลี้ยง)

รายงานวิจัยฉบับสมบูรณ์

โครงการ: การใช้คอมพิวเตอร์ เพื่อศึกษาความเป็นไปได้ของการเกิดปรากฏการณ์
ควอนตัมคอนไฟน์สตาร์ก ในหลุมควอนตัมของแหล่งกำเนิดแสงชนิดสารกึ่งตัวนำ

โดย ดร.ภัทร อัยรักษ์ มหาวิทยาลัยสงขลานครินทร์
รศ.บุญเหลือ พงศ์คารา มหาวิทยาลัยสงขลานครินทร์ (นักวิจัยพี่เลี้ยง)

สนับสนุนโดยสำนักงานกองทุนสนับสนุนการวิจัย

(ความเห็นในรายงานนี้เป็นของผู้วิจัย สกว.ไม่จำเป็นต้องเห็นด้วยเสมอไป)

กิตติกรรมประกาศ

งานวิจัยชิ้นนี้สำเร็จได้โดยการสนับสนุนจากหลายฝ่าย โดยเฉพาะอย่างยิ่งสำนักงานกองทุนสนับสนุนการวิจัยซึ่งสนับสนุนทุนวิจัยอย่างพอเพียงและต่อเนื่อง นอกจากนี้ยังได้รับการสนับสนุนทางวิชาการจาก Dr. D.W.E. Allsopp อาจารย์จากภาควิชา Electronic & Electrical Engineering University of Bath สหราชอาณาจักรและ Dr. A.J. Vickers อาจารย์จากภาควิชา Electronic Systems Engineering University of Essex สหราชอาณาจักร นอกจากนี้ ผู้ที่ขาดมิได้ในงานวิจัยชิ้นนี้คือ รศ.บุญเหลือ พงศ์ดารา นักวิจัยพี่เลี้ยงและผู้สนับสนุนการขอทุนครั้งนี้ ผู้วิจัยรู้สึกซาบซึ้งในความช่วยเหลือของทุกท่านเป็นอย่างสูง

ภัทร อัยรักษ์

สารบัญ

	หน้า
กิตติกรรมประกาศ	a
สารบัญ	b
Abstract	1
บทคัดย่อ	2
บทนำ	3
วิธีการทดลอง	5
ผลการทดลอง	6
บทวิจารณ์	14
เอกสารอ้างอิง	16
Output จากโครงการวิจัยที่ได้รับทุนจาก สกว.	17
ภาคผนวก ก. การใช้งานโปรแกรมพร้อม Screen Shots ตัวอย่าง	18
ภาคผนวก ข. โครงสร้างของแฟ้มข้อมูล Input และ Output	21
ภาคผนวก ค. Class และ Main Program สำหรับโปรแกรมบน UNIX	23
ภาคผนวก ง. Class และ Main Program สำหรับโปรแกรมบน PC	36
ภาคผนวก จ. ผลงานในการประชุมวิชาการ	46
ภาคผนวก ฉ. Manuscript	49

Abstract

Project Code: PDF/26/2544

Project Title: A computer simulation search for the quantum confined Stark effect in a semiconductor light source with the quantum well.

Investigator: Dr. Pattara Aiyarak, Prince of Songkla University

E-mail Address: apattara@ratree.psu.ac.th

Project Period: 1 July 2001 – 31 December 2003

A computer modeling was created in order to find a suitable structure for semiconductor light emitters. The model was designed to cope with an ability to solve drift-diffusion and Schrödinger equations self-consistently; therefore, the semiconductor devices with quantum wells can be simulated. The main aim of this work is to find a suitable structure for a semiconductor light source which has the ability to emit the multi-wavelength light using the quantum confined Stark effect. There are two programs on this work. The self-consistent model was run on the Sun Workstation using UNIX system because of its better performance and another model was able to solve only the drift-diffusion problem and created on Visual C++ platform which runs on a general personal computer. However, the object-oriented programming style was used on both programs. However, the later model is easier to use because of the nature of the Visual C++ application. Possible structures of the multi-wavelength light emitters were simulated and a suitable structure was found and discussed. The structure is an AlGaAs system with a single quantum well.

Keywords: quantum confined Stark's effect, multiple wavelength light emitter, simulation, self-consistence

บทคัดย่อ

รหัสโครงการ: PDF/26/2544

โครงการ: การใช้คอมพิวเตอร์ เพื่อศึกษาความเป็นไปได้ของการเกิดปรากฏการณ์ควอนตัมคอนไฟน์เมนต์ในหลุมควอนตัมของแหล่งกำเนิดแสงชนิดสารกึ่งตัวนำ

นักวิจัย: ดร.ภัทร อัยรักษ์ มหาวิทยาลัยสงขลานครินทร์

E-mail Address: apattara@ratree.psu.ac.th

ระยะเวลาโครงการ: 1 กรกฎาคม 2544 – 31 ธันวาคม 2546

โปรแกรมคอมพิวเตอร์เพื่อหาโครงสร้างที่เหมาะสมของแหล่งกำเนิดแสงชนิดสารกึ่งตัวนำถูกเขียนขึ้นเพื่อหาผลเฉลยแบบ self-consistent ระหว่างสมการ drift-diffusion และ Schrödinger โปรแกรมดังกล่าวจึงสามารถหาผลเฉลยของโครงสร้างของสารกึ่งตัวนำที่มีหลุมควอนตัมอยู่ภายในได้ วัตถุประสงค์หลักของงานวิจัยนี้คือการหาโครงสร้างที่เหมาะสมสำหรับแหล่งกำเนิดแสงชนิดสารกึ่งตัวนำที่สามารถเปล่งแสงได้หลายความยาวคลื่นโดยใช้หลักการของปรากฏการณ์ควอนตัมคอนไฟน์เมนต์ โปรแกรมที่ได้มีสองรูปแบบคือโปรแกรมที่สามารถแก้ปัญหาแบบ self-consistent ได้ซึ่งเป็นโปรแกรมที่เขียนบน Sun Workstation ที่ใช้ UNIX เป็นระบบปฏิบัติการเนื่องจากความสามารถที่สูง และเหมาะสมกว่า และอีกโปรแกรมหนึ่งคือโปรแกรมที่แก้ปัญหาเฉพาะสมการ drift-diffusion ซึ่งเขียนโดยใช้ Visual C++ และทำงานบนเครื่องคอมพิวเตอร์ส่วนบุคคลทั่วไปซึ่งใช้งานง่ายสำหรับบุคคลทั่วไป อย่างไรก็ตามทั้งสองโปรแกรมใช้การโปรแกรมแบบเชิงวัตถุ (object oriented programming) เพื่อความง่ายและสะดวกในการแก้ไขภายหลัง ผลเฉลยของโครงสร้างของแหล่งกำเนิดแสงที่น่าจะเป็นไปได้หลายโครงสร้าง ซึ่งทั้งหมดเป็นโครงสร้างของ AlGaAs ที่มีหลุมควอนตัมอยู่ภายในได้ถูกนำมาวิเคราะห์ และโครงสร้างที่เหมาะสมที่สุดได้ถูกเลือกขึ้นมาเพื่อศึกษาต่อไป

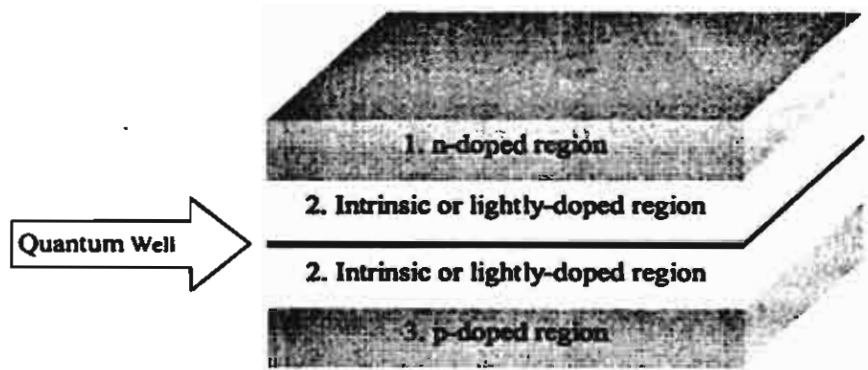
คำหลัก: quantum confined Stark's effect, multiple wavelength light emitter, simulation, self-consistence

บทนำ

การพัฒนาแหล่งกำเนิดแสงที่สามารถเปล่งแสงได้หลายความยาวคลื่นเป็นสิ่งที่จำเป็นในการเพิ่มความเร็วในการสื่อสาร ซึ่งการเพิ่มความเร็วดังกล่าวอาศัยหลักการส่งสัญญาณแสงหลายความยาวคลื่นในเวลาเดียวกัน (Wavelength Division Multiplexing, WDM) การศึกษาความเป็นไปได้ของการเกิดปรากฏการณ์ควอนตัมคอนไฟน์เมนต์ในหลุมควอนตัมของแหล่งกำเนิดแสงชนิดสารกึ่งตัวนำ นำไปสู่การสร้างแหล่งกำเนิดแสงชนิดสารกึ่งตัวนำชนิดใหม่ที่สามารถนำมาประยุกต์ใช้ได้กับการสื่อสารแบบ WDM การทดลองสร้างและการทดสอบสิ่งประดิษฐ์สารกึ่งตัวนำชนิดดังกล่าวจำเป็นอย่างยิ่งที่จะต้องใช้เครื่องมือที่มีราคาสูง การใช้คอมพิวเตอร์เพื่อช่วยในการศึกษาจึงเป็นทางออกหนึ่งในการศึกษา นอกจากนี้โปรแกรมที่ถูกเขียนขึ้นเพื่อทดสอบสามารถนำไปใช้ศึกษาสิ่งประดิษฐ์สารกึ่งตัวนำอื่นๆ ได้

โปรแกรมสามารถแบ่งอย่างคร่าวๆ ได้สองส่วนคือส่วนที่หนึ่งเป็นส่วนที่เรียกว่า Drift-Diffusion Model ซึ่งเป็นแบบจำลองเพื่อใช้ในการศึกษาการเคลื่อนที่ของอิเล็กตรอน และโฮลของ Bulk Material ซึ่ง Drift-Diffusion Model นี้คือการหาผลเฉลยของสมการ Poisson และ Continuity Equations สำหรับส่วนที่สองคือ Quantum Model โดยในส่วนนี้เป็นการหาผลเฉลยของสมการ Schrödinger เพื่อศึกษาการเคลื่อนที่ของอิเล็กตรอน และโฮลในหลุมควอนตัม เมื่อนำโปรแกรมสองส่วนนี้มารวมกันก็จะสามารถศึกษาระดับพลังงาน และการเคลื่อนที่ของของสิ่งประดิษฐ์สารกึ่งตัวนำที่มีหลุมควอนตัมอยู่ภายในได้

โครงสร้างของแหล่งกำเนิดแสงชนิดสารกึ่งตัวนำที่จะนำมาทดสอบนี้ประกอบขึ้นจาก $\text{Al}_x\text{Ga}_{1-x}\text{As}$ และ GaAs หลายชั้นประกอบกันโดยมีหลุมควอนตัมอยู่ตรงกลาง ความพิเศษของโครงสร้างนี้คือการมีกำแพงศักย์กั้นระหว่างสารกึ่งตัวนำชนิด n และหลุมควอนตัม โดยเมื่อแหล่งกำเนิดแสงถูกไบแอสตาม โวลจากสารกึ่งตัวนำชนิด p จะเคลื่อนที่เข้ามายังหลุมควอนตัมในขณะที่อิเล็กตรอนจากสารกึ่งตัวนำชนิด n ไม่สามารถเคลื่อนที่เข้ามายังหลุมควอนตัมได้เนื่องจากมีกำแพงศักย์กั้นอยู่ โดยในช่วงนี้จะยังไม่มีแสงออกมา จนกระทั่งสารกึ่งตัวนำชนิด n ได้รับพลังงานโดยการจ่ายสนามไฟฟ้าแบบตั้งฉากกับแนวของโครงสร้างเพื่อทำให้อุณหภูมิของอิเล็กตรอนสูงขึ้น จนสามารถเอาชนะกำแพงศักย์ และเคลื่อนที่เข้าไปยังหลุมควอนตัมและเปล่งแสงออกมาได้ ในส่วนของการเปลี่ยนความยาวคลื่นสามารถทำได้โดยการปรับขนาดของการไบแอสตามเพื่อทำให้มีสนามไฟฟ้าตกคร่อมบนหลุมควอนตัมขนาดต่างๆ กัน เป็นผลให้ความยาวคลื่นที่ได้ต่างกัน ซึ่งเกิดจากปรากฏการณ์ควอนตัมคอนไฟน์เมนต์



ภาพที่ 1: โครงสร้างที่นำมาศึกษา

ภาพที่ 1 แสดงโครงสร้างของสิ่งประดิษฐ์สารกึ่งตัวนำที่ต้องการนำมาศึกษา โดยเปลี่ยนค่าความหนา การเจือสาร และชนิดของสารแต่ละชั้น โดยกำหนดให้สารแต่ละชั้นคือ $\text{Al}_x\text{Ga}_{1-x}\text{As}$ เนื่องจากสิ่งประดิษฐ์ที่สร้างมาจากสารกึ่งตัวนำดังกล่าว สามารถสร้างได้ง่าย

วิธีการทดลอง

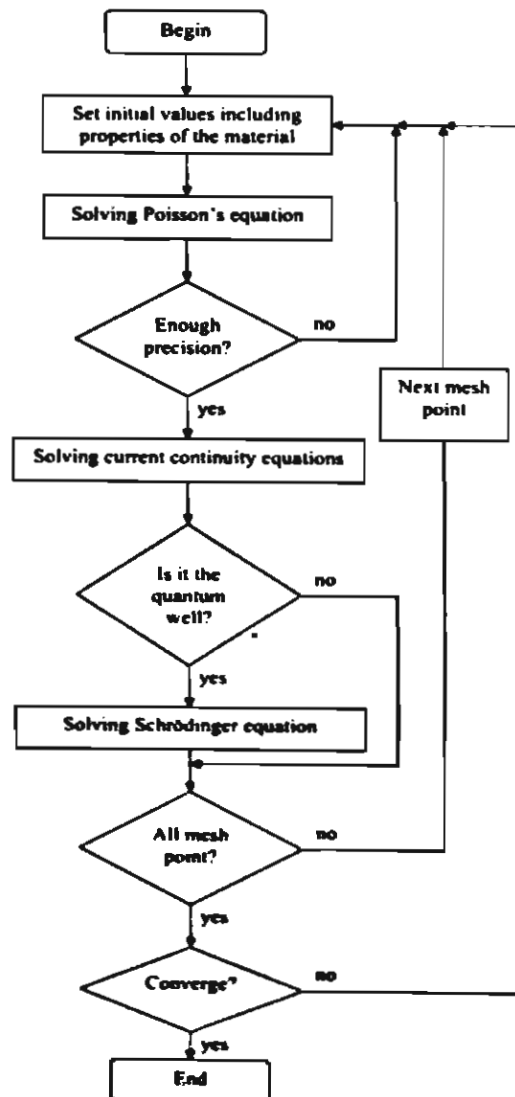
1. ออกแบบโครงสร้างของโปรแกรมโดยสร้าง Flowcharts ของระบบทั้งหมด
2. ออกแบบและสร้างโปรแกรมย่อยต่างๆ ตามรูปแบบของ Flowcharts ที่ได้ออกแบบในข้อ 1.
3. นำโปรแกรมย่อยทั้งหมดมาทดสอบการทำงานเพื่อก่อนที่จะนำโปรแกรมย่อยทั้งหมดมารวมกัน
4. ทดสอบโปรแกรมที่ได้โดยทำแบบจำลองอุปกรณ์อิเล็กทรอนิกส์พื้นฐาน เช่น ไดโอด และ ทรานซิสเตอร์ เพื่อทดสอบการทำงานของโปรแกรม
5. ทำแบบจำลองของอุปกรณ์ที่ต้องการศึกษาความเป็นไปได้ในการเกิด ปรากฏการณ์ควอนตัมคอนไฟน์สตาร์ก

ผลการทดลอง

โปรแกรมได้ถูกเขียนขึ้นด้วย Visual C++ ซึ่งถูกแบ่งออกเป็นสองส่วนคือ Drift-Diffusion Model และ Quantum Model ซึ่ง Drift-Diffusion Model เป็นการแก้สมการ Poisson และ Continuity ส่วน Quantum Model เป็นการแก้สมการ Schrödinger ซึ่งในการหาผลเฉลยที่มีความซับซ้อนไม่สามารถทำได้โดยวิธีการวิเคราะห์ ซึ่งการหาผลเฉลยวิธีหนึ่งคือการนำคอมพิวเตอร์เข้ามาช่วย การแก้สมการจะใช้ระเบียบวิธี Finite Difference ซึ่งในการหาอนุพันธ์ของสมการใดๆ สามารถทำได้ดังสมการ

$$\frac{\partial f}{\partial x} = \lim_{h \rightarrow 0} \frac{f\left(x + \frac{h}{2}\right) - f\left(x - \frac{h}{2}\right)}{h} \quad (1)$$

Flowchart ของโปรแกรมที่เกิดจากการนำเอาทั้งสอง Models มารวมกันสามารถแสดงได้ย่อกว่าๆ ดังนี้



ภาพที่ 2 แสดง Flowchart ของโปรแกรม

สมการปัวซอง (Poisson's Equation)

สมการปัวซอง (Poisson's Equation) ใช้ในการแก้ปัญหาของสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งเป็นสมการแสดงความสัมพันธ์ระหว่างประจุและสนามไฟฟ้าจากสมการที่ (2) แสดงให้เห็นถึงความสัมพันธ์ระหว่างสนามไฟฟ้า (Electric Field) และความหนาแน่นของประจุไฟฟ้า (Electric Charge Density) สามารถเขียนเป็นสมการได้ว่า^[1]

$$\vec{\nabla} \cdot \vec{D} = \rho(x, y, z) \quad (2)$$

สมการที่ (2) เรียกว่า สมการปัวซองในสามมิติ โดยที่ $\vec{D}$ คือ เวกเตอร์การกระจัด และ $\rho(x, y, z)$ คือ ผลรวมของความหนาแน่นของประจุไฟฟ้า อย่างไรก็ตามแต่สำหรับในที่นี้จะแสดงตัวอย่างวิธีการหาผลเฉลยเฉพาะสมการปัวซองในหนึ่งมิติ ความสัมพันธ์ระหว่าง $\vec{D}$ และสนามไฟฟ้า $\vec{E}$ สามารถเขียนได้ตามสมการที่ (3)

$$\vec{D} = \epsilon \vec{E} \quad (3)$$

ค่าของ ϵ ตามสมการ (3) คือ ค่าคงที่ไดอิเล็กตริก (Permittivity) ของสาร ในกรณีของสารกึ่งตัวนำชนิดนี้ ค่าความหนาแน่นของประจุไฟฟ้าเขียนเป็นสมการได้ว่า

$$\rho = q(p - n + N_D - N_A) \quad (4)$$

โดยที่ q คือขนาดของประจุไฟฟ้า p และ n คือค่าความหนาแน่นของประจุโฮลและอิเล็กตรอน ค่า N_D และ N_A คือผลรวมของอะตอมผู้ให้ (Donor) และอะตอมผู้รับ (Acceptor) ต่อตารางเซนติเมตรแล้วแทนสมการ (3) และสมการ (4) ลงในสมการ (2) สามารถเขียนใหม่ได้ว่า

$$\vec{\nabla} \cdot \epsilon \vec{E} = q(p - n + N_D - N_A) \quad (5)$$

ซึ่ง สนามไฟฟ้า $\vec{E}$ สามารถเขียนในรูป ศักย์ไฟฟ้า ϕ เขียนเป็นสมการได้ว่า

$$\vec{E} = -\vec{\nabla} \phi \quad (6)$$

แทนสมการ (6) ลงใน สมการ (5) จะได้ความสัมพันธ์ว่า

$$\vec{\nabla} \cdot (\epsilon \vec{\nabla} \phi) = \epsilon \nabla^2 \phi + \vec{\nabla} \epsilon \cdot \vec{\nabla} \phi = q(n - p + N_A - N_D) \quad (7)$$

สมการความต่อเนื่อง (Continuity Equations)

สมการความต่อเนื่องเป็นสมการที่สามารถแสดงลักษณะการเปลี่ยนแปลงของจำนวนพาหะนำประจุในบริเวณใดบริเวณหนึ่งของก้อนสารกึ่งตัวนำได้และสามารถเขียนสมการแสดงความสัมพันธ์ได้ว่า

$$\vec{\nabla} \times \vec{H} = \frac{\partial \vec{D}}{\partial t} + \vec{J}_{cond} \quad (8)$$

โดยที่ $\vec{H}$ คือ สนามแม่เหล็ก และ $\vec{J}_{cond}$ คือความหนาแน่นของกระแสตัวนำ จะนำตัวดำเนินการ $\vec{\nabla} \cdot$ ลงในสมการ (8) และแทนสมการ (2) ลงใน สมการ (8) เช่นกัน ซึ่งสามารถเขียนสมการแสดงความสัมพันธ์ได้ว่า

$$\vec{\nabla} \cdot \vec{\nabla} \times \vec{H} = \frac{\partial \rho}{\partial t} + \vec{\nabla} \cdot \vec{J}_{cond} = 0 \quad (9)$$

ค่าความหนาแน่นของกระแสตัวนำสามารถเขียนให้อยู่ในรูปองค์ประกอบของ $\vec{J}_p$ ซึ่งก็คือความหนาแน่นของกระแสตัวนำในโฮล และในรูปองค์ประกอบของ $\vec{J}_n$ ซึ่งก็คือความหนาแน่นของกระแสตัวนำในอิเล็กตรอน ประจุทั้งหมดในสารกึ่งตัวนำยกเว้นอิเล็กตรอนและโฮลอิสระ ถูกสมมุติให้ ไม่มีการเปลี่ยนแปลงตามเวลา ดังนั้น สมการ (9) สามารถเขียนใหม่ได้ว่า

$$\vec{\nabla} \cdot (\vec{J}_p + \vec{J}_n) + \frac{\partial}{\partial t} (p - n) = 0 \quad (10)$$

สมการที่ (10) สามารถเขียนสมการความต่อเนื่องได้เป็น 2 สมการ โดยที่ กำหนดให้ปริมาณ R คือค่าคงที่ค่าหนึ่ง ดังนั้นสมการที่ (10) สามารถเขียนให้อยู่ในรูปของ R ได้ว่า

$$\vec{\nabla} \cdot \vec{J}_n - q \frac{\partial n}{\partial t} = qR \quad (11)$$

$$\vec{\nabla} \cdot \vec{J}_p - q \frac{\partial p}{\partial t} = -qR \quad (12)$$

ซึ่งปริมาณ R คือฟังก์ชันที่เป็นผลรวมของปริมาณสุทธิที่เกิดจากการเกิดขึ้นใหม่ (Generation) และการรวมตัว (Recombination) ระหว่างอิเล็กตรอนและโฮล กำหนดให้ค่า R มีค่าเป็นบวกเมื่อมีการเกิดขึ้นใหม่ (Generation) ระหว่างอิเล็กตรอนและโฮล และค่า R มีค่าเป็นลบ ก็ต่อเมื่อการเกิดการรวมตัว (Recombination) ระหว่างอิเล็กตรอนและโฮล

สมการการไหลของกระแส (Current Density Equations)

สมการความหนาแน่นของกระแสของอิเล็กตรอนและโฮล ประกอบด้วยค่าความหนาแน่นของกระแสอันเนื่องมาจากสนามไฟฟ้า กล่าวคือ การที่เราให้สนามไฟฟ้า E และสนามไฟฟ้านี้จะออกแรงกระทำบนอิเล็กตรอน และโฮลทำให้เกิดการเคลื่อนที่ของพาหะประจุในสนามไฟฟ้า และความหนาแน่นของกระแสอันเนื่องมาจากเกิดการแพร่เนื่องจากความแตกต่างกันของความหนาแน่นประจุ ซึ่งสามารถอธิบายได้โดยใช้สมการที่ (13)

$$\vec{J}_n = -q\mu_n \vec{\nabla} \Psi_n + qD_n \vec{\nabla} n \quad (13)$$

$$\vec{J}_p = -q\mu_p \vec{\nabla} \Psi_p - qD_p \vec{\nabla} p \quad (14)$$

โดยที่ μ_n และ μ_p คือสภาพความคล่องตัว (Mobility) ของอิเล็กตรอนและโฮล D_n และ D_p คือค่าสัมประสิทธิ์ของการแพร่ (Diffusion Coefficients) ของอิเล็กตรอนและโฮล

สำหรับค่า Ψ_n และ Ψ_p คือศักย์ไฟฟ้า (Electrostatic Potentials) ของอิเล็กตรอนและโฮล ซึ่งเขียนเป็นสมการได้ว่า^[2]

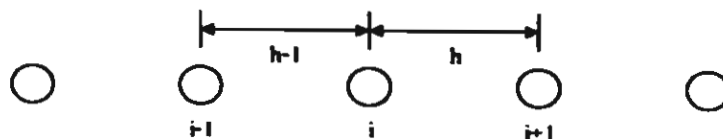
$$\Psi_n = \phi + \frac{\chi - \chi_s}{q} + \frac{k_s T}{q} \ln \left(\frac{N_c}{N_v} \right) \quad (15)$$

$$\Psi_p = \phi + \frac{\chi - \chi_s}{q} + \left(\frac{E_g - E_v}{q} \right) + \frac{k_s T}{q} \ln \left(\frac{N_v}{N_c} \right) \quad (16)$$

โดยที่ χ คือค่าสภาพอ่อนไหวทางไฟฟ้าของอิเล็กตรอน (Electron Affinity), E_g คือช่องว่างของแถบพลังงาน (Energy Bandgap), k_s คือคงที่ของโบสมานด์ N_c และ N_v คือ ความหนาแน่นสถานะยังผล (Effective Density of States) ในแถบการนำและแถบวาเลนซ์ และตัวห้อย r คือค่าดัชนีที่แสดงถึงตัวแปรที่ใช้อ้างอิงของสาร (Reference Material)^[3]

ระเบียบวิธี Discretization

สมการทางฟิสิกส์ปริมาณส่วนใหญ่จะเป็นฟังก์ชันที่มีความต่อเนื่องการใช้สมการทางคณิตศาสตร์สำหรับหาค่าระหว่างจุดต่อจุดมีความจำเป็นเป็นอย่างมากในการออกแบบสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งจะส่งผลให้หาค่าตัวแปรต่างๆ ได้ง่าย และทุกๆ จุดบนสิ่งประดิษฐ์มีความต่อเนื่องกันในที่นี้แสดงแค่หนึ่งมิติ คือเฉพาะแกน X เท่านั้น ดังภาพที่ 3 สามารถช่วยให้การคำนวณค่าตัวแปรต่างๆ มีความแม่นยำมากยิ่งขึ้นอันได้แก่ ค่าความหนาแน่นของโฮลและอิเล็กตรอน ศักย์ไฟฟ้า เป็นต้น



ภาพที่ 3: แสดงการกำหนดจุดในสิ่งประดิษฐ์สารกึ่งตัวนำใน 1 มิติ

สำหรับ Quantum Model การหาระดับพลังงานของหลุมควอนตัมชนิดสี่เหลี่ยมสามารถหาได้จากการแก้สมการ Schrödinger ซึ่งระดับพลังงานสามารถหาได้จากสมการ^[4]

$$E_n = -\Delta E_c (1 - \alpha_n^2) \quad (17)$$

โดยที่ α_n หาได้จากสมการ

$$\gamma \alpha_n = (n-1)\pi + 2 \cos^{-1} \alpha_n \quad (18)$$

ซึ่ง

$$\gamma = \sqrt{\frac{2m\Delta E_c L_z^2}{\hbar^2}} \quad (19)$$

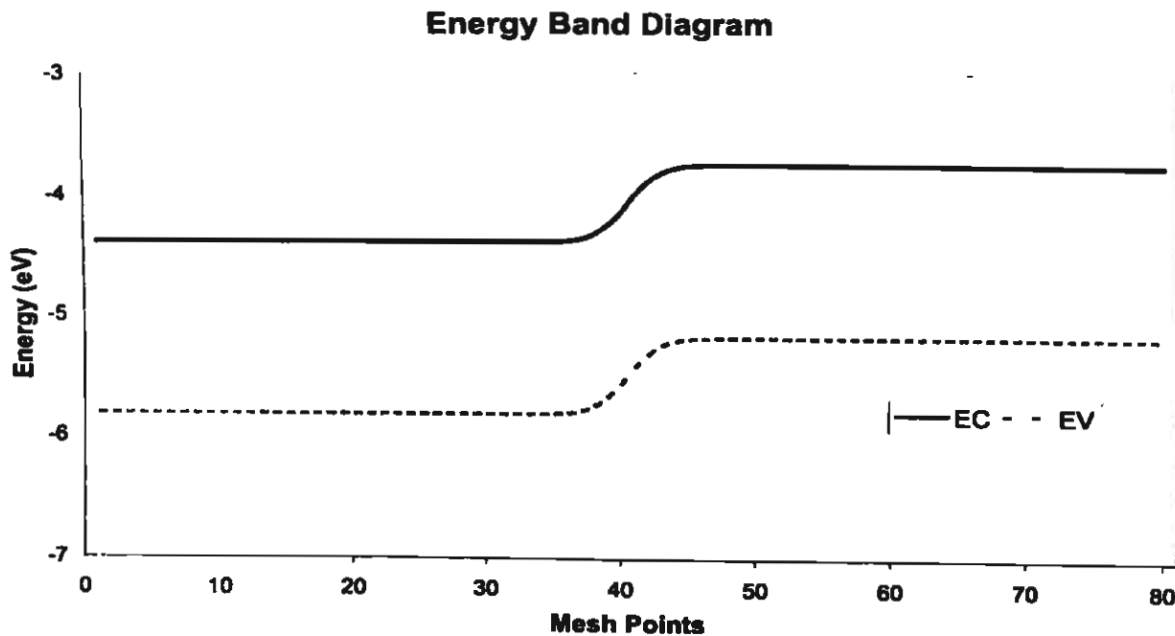
และ Electron Concentration ใน 2 มิติ (2DEG) หาได้จากสมการ

$$n_{2DEG} = \sum_n \frac{mk_B T}{\pi \hbar^2} \ln \left[1 + e^{\frac{E_{FV} - E_n}{k_B T}} \right] |\Psi_n(z)|^2 \quad (20)$$

และในทำนองเดียวกัน

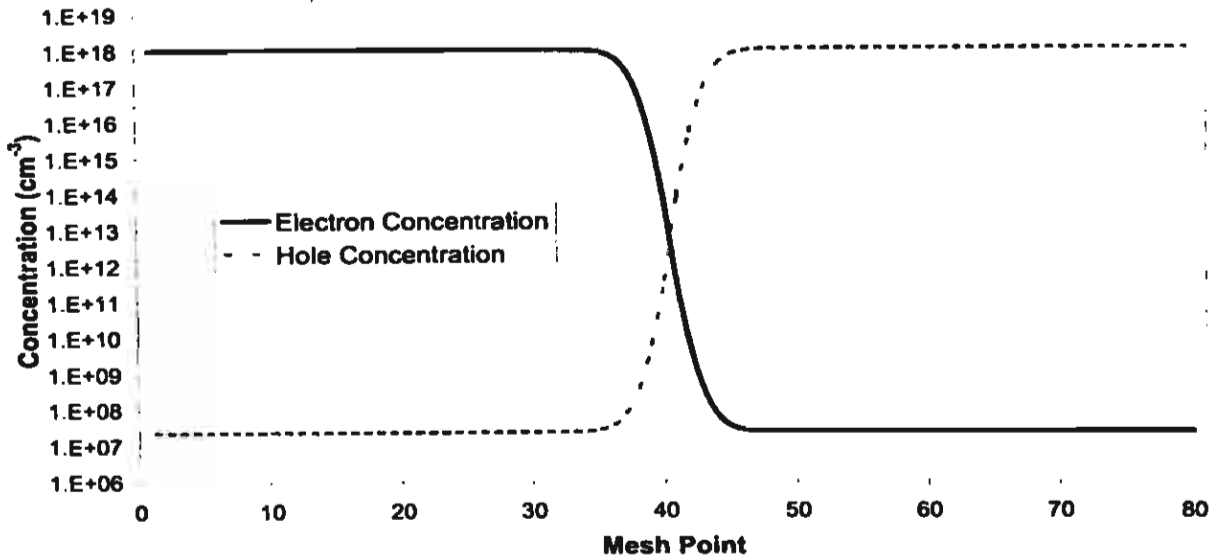
$$p_{2DHG} = \sum_n \frac{mk_B T}{\pi \hbar^2} \ln \left[1 + e^{\frac{E_n - E_{FP}}{k_B T}} \right] |\Psi_n(z)|^2 \quad (21)$$

การนำ Drift-Diffusion Model มาทดสอบการทำงานโดยการนำ Diode ซึ่งเป็นอุปกรณ์สารกึ่งตัวนำพื้นฐานมาทดสอบ โดย Diode ที่นำมาทดสอบคือ GaAs p-n junction และมี Doping Concentration เท่ากับ 10^{18} cm^{-3} ทั้งด้าน p และ n ผลเฉลยที่ได้ดังแสดงในภาพที่ 3-4 คือ Energy Band Diagram และ Doping Concentrations



ภาพที่ 3: แสดงค่า E_c และ E_v ซึ่งเป็นผลเฉลยจากไดโอดชนิด p-n Junction

Carrier Concentration



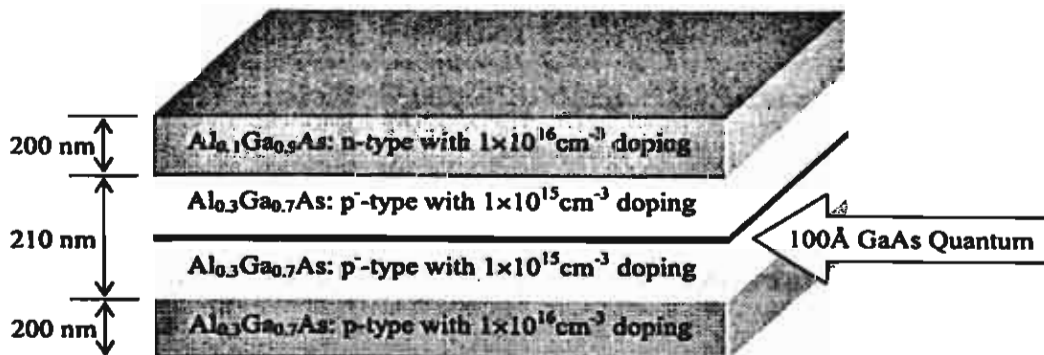
ภาพที่ 4: แสดงค่าความหนาแน่นของอิเล็กตรอนและโฮล
ซึ่งเป็นผลเฉลยจากไดโอดชนิด p-n Junction

โดยที่ในกรณีนี้ 1 Mesh Point มีค่าเท่ากับ 5×10^{-7} cm และจากการคำนวณหา Depletion Region โดยสมการ

$$w_0 = x_n + x_p = \left[\frac{2\epsilon}{q} \left(\frac{N_A + N_D}{N_A N_D} \right) V_b \right]^{1/2} \quad (22)$$

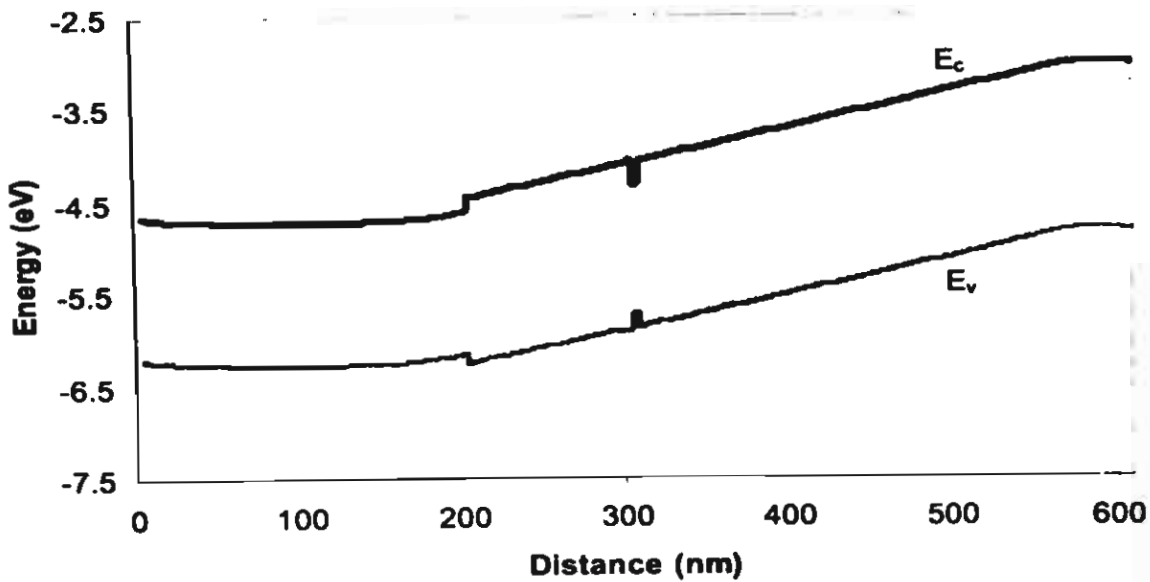
จะได้ว่าค่า Depletion Region มีค่าเท่ากับ 4.30×10^{-5} cm และจากการคำนวณจาก Energy Band Diagram พบว่าค่า Depletion Region มีค่าเท่ากับ $4.0 \pm 0.5 \times 10^{-5}$ cm ซึ่งค่าที่ได้ตรงกับที่คำนวณได้จากสมการ

ส่วนตัวอย่างการจำลองแบบของสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งมีหลุมควอนตัมด้วยจะทดสอบโดยโครงสร้างดังภาพที่ 5

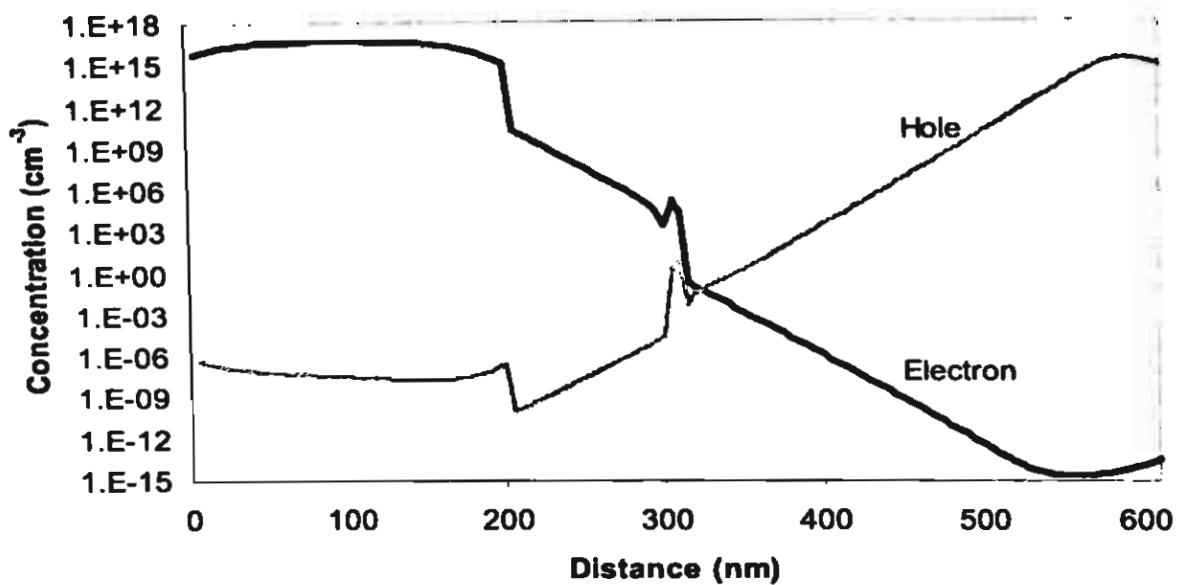


ภาพที่ 5 แสดงโครงสร้างของสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งมีหลุมควอนตัม

ผลเฉลยในกรณี Equilibrium แสดงในภาพที่ 5-6 คือ Energy Band Diagram และ Doping Concentrations ของสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งมีหลุมควอนตัมอยู่ภายใน

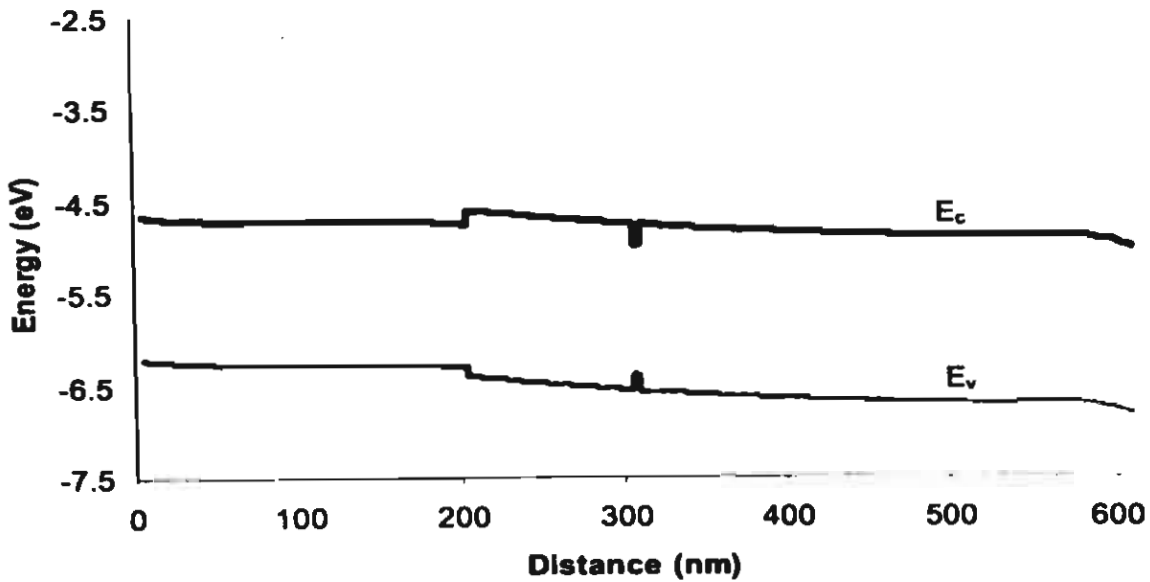


ภาพที่ 5: แสดงค่า E_c และ E_v ซึ่งเป็นผลเฉลยในกรณี Equilibrium จากสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งมีหลุมควอนตัมอยู่ภายใน

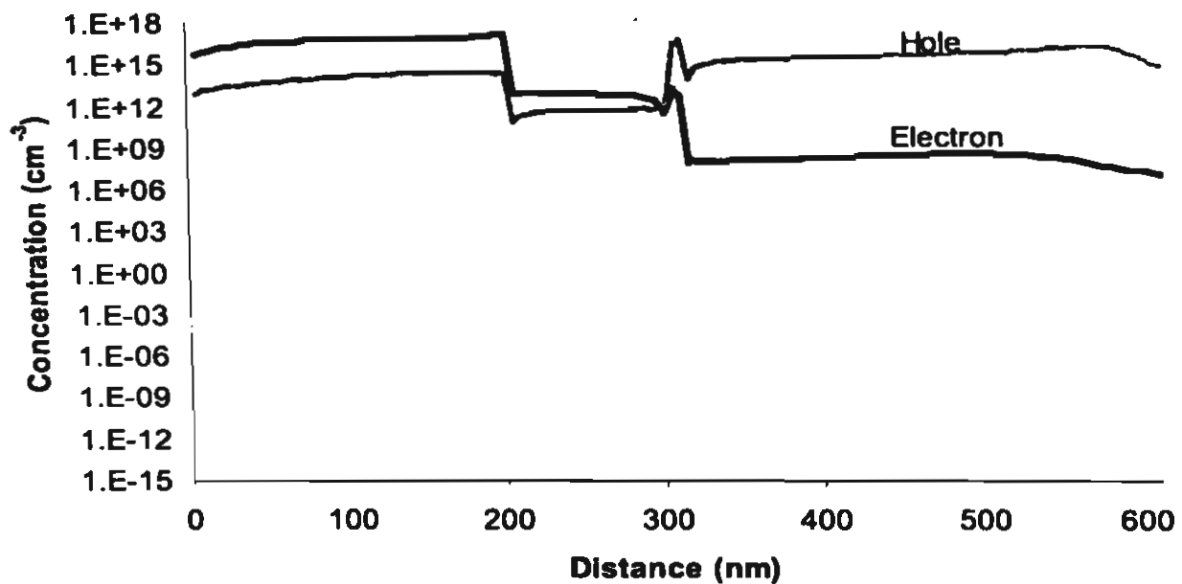


ภาพที่ 6: แสดงค่าความหนาแน่นของอิเล็กตรอนและโฮล ซึ่งเป็นผลเฉลยในกรณี Equilibrium จากสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งมีหลุมควอนตัมอยู่ภายใน

และผลเฉลยในกรณีที่มีการไบแอสตรงขนาด 2 V แสดงในภาพที่ 7-8



ภาพที่ 7: แสดงค่า E_c และ E_v ซึ่งเป็นผลเฉลยในกรณี 2 Volt. Forward Bias จากสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งมีหลุมควอนตัมอยู่ภายใน



ภาพที่ 8: แสดงค่าความหนาแน่นของอิเล็กตรอนและโฮล ซึ่งเป็นผลเฉลยในกรณี 2 Volt. Forward Bias จากสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งมีหลุมควอนตัมอยู่ภายใน

บทวิจารณ์

สรุปผลการทดลอง

ผลการทดสอบโปรแกรมดังที่ได้แสดงไปในหัวข้อผลการทดลอง แสดงให้เห็นถึงความสามารถในการหาผลเฉลยของโปรแกรม โดยโปรแกรมสามารถทำงานได้ดี และมีเสถียรภาพสูงเมื่อทำการหาผลเฉลยในส่วนของสิ่งประดิษฐ์สารกึ่งตัวนำที่ไม่มีหลุมควอนตัม หรือไม่มีความจำเป็นในการใช้สมการ Schrödinger ในการหาผลเฉลย อย่างไรก็ตาม ความเร็วในการหาผลเฉลยยังไม่สูง หากโปรแกรมทำงานบนเครื่องคอมพิวเตอร์ที่มีความเร็วหรือมีความจุของหน่วยความจำต่ำ นอกจากนี้หากต้องการหาผลเฉลยในกรณี Steady-State ก็มีการไบแอสด้วยความต่างศักย์ค่าต่างๆ จะทำให้เวลาในการหาผลเฉลยสูงขึ้น อย่างไรก็ตามหากต้องการหาผลเฉลยในกรณีที่โครงสร้างไม่ซับซ้อนก็สามารถหาได้ในระยะเวลาอันรวดเร็ว

การแก้ปัญหาในเรื่องเวลาในการหาผลเฉลย สามารถทำได้โดยเปลี่ยน ชนิดของเครื่องคอมพิวเตอร์ โดยเปลี่ยนมาเป็นเครื่องคอมพิวเตอร์ชนิด Workstation โดยในที่นี้จะใช้เครื่อง Sun Workstation ซึ่งทำงานบนระบบปฏิบัติการ UNIX ซึ่งทำให้ความเร็วในการหาผลเฉลยเร็วขึ้น อย่างไรก็ตามหากโครงสร้างมีความซับซ้อนสูง ระยะเวลาในการหาผลเฉลยก็นานขึ้น แต่เนื่องจากธรรมชาติของเครื่อง Workstation ซึ่งทำงานตลอดเวลา ทำให้สามารถทิ้งโปรแกรมให้หาผลเฉลยได้เป็นเวลานาน

สำหรับการหาผลเฉลยในกรณีที่ต้องใช้การแก้สมการ Schrödinger เข้าร่วมกับสมการ Drift-Diffusion โปรแกรมสามารถหาผลเฉลยได้เกือบทุกกรณี อย่างไรก็ตามเนื่องจากปัญหาการลู่เข้า (Convergence) ของผลเฉลยบางกรณี โดยเฉพาะอย่างยิ่งในกรณี Steady-State โปรแกรมใช้เวลาในการหาผลเฉลยนานมาก หรือในบางครั้งก็ไม่สามารถหาผลเฉลยได้เลย เนื่องจากผลเฉลยไม่ลู่เข้าหาคำตอบ

อย่างไรก็ตามโปรแกรมสามารถใช้งานได้ดีในระดับหนึ่ง โดยเฉพาะหากไม่ต้องการหาผลเฉลยจากโครงสร้างที่ซับซ้อนมากและไม่มีจำเป็นต้องใช้สมการ Schrödinger ในการหาผลเฉลย โปรแกรมซึ่งทำงานบนเครื่องคอมพิวเตอร์ส่วนบุคคล จะใช้งานได้ง่ายและเหมาะสมสำหรับผู้ที่ต้องการหาผลเฉลยที่ไม่ซับซ้อนมากนัก และไม่มีความรู้ในการแก้ไขโปรแกรม เนื่องจากตัวโปรแกรมเป็น Graphical User Interface (GUI) ซึ่งใช้งานได้ง่าย

ข้อเสนอแนะ

เนื่องจากโปรแกรมไม่สามารถแก้ปัญหา Self-Consistence ได้ทุกกรณี และความเร็วในการหาผลเฉลยที่ซับซ้อนยังไม่สูงมาก ทำให้การแก้ปัญหาดังกล่าวเป็นสิ่งที่ควรปรับปรุง อย่างไรก็ตามเนื่องจากธรรมชาติของปัญหา ทำให้บางปัญหาไม่สามารถหาผลเฉลยได้

ในส่วน of โปรแกรมที่เป็น GUI ผู้ใช้ยังจำเป็นต้องเขียนแฟ้มโครงสร้างเองซึ่งอาจทำให้มีความผิดพลาดได้ง่าย หากสามารถพัฒนาโปรแกรมในส่วนของการสร้างแฟ้มโครงสร้างให้ทำงานได้ง่ายขึ้น จะเป็นประโยชน์สำหรับผู้ใช้นอกจากส่วนของแฟ้มโครงสร้างแล้ว ในส่วนของผลเฉลย

ณ ปัจจุบันยังจำเป็นต้องใช้โปรแกรมประเภท Spreadsheet เข้าช่วยในการสร้างกราฟ ซึ่งหากโปรแกรมที่เขียนขึ้นสามารถสร้างกราฟได้โดยตรง จะทำให้ง่ายในการใช้งานมากขึ้น หากโปรแกรมมีความสมบูรณ์เพียงพอ ผู้วิจัยคาดว่าจะสามารถเผยแพร่ได้ทั้งในระดับประเทศ และนานาชาติ เพื่อใช้ทั้งในการเรียนการสอน และงานวิจัยทางสิ่งประดิษฐ์สารกึ่งตัวนำ

นอกจากการพัฒนาในส่วนที่กล่าวถึงแล้วการทำให้โปรแกรมสามารถแก้ปัญหาโครงสร้างเป็นโครงสร้างหลายมิติได้ จะทำให้สามารถแก้ปัญหาโครงสร้างที่ซับซ้อนขึ้นได้ รวมถึงสามารถหาผลเฉลยของสิ่งประดิษฐ์สารกึ่งตัวนำที่มีส่วนประกอบของจุด หรือเส้นควอนตัม (Quantum Dot or Quantum Line) ได้

เอกสารอ้างอิง

- [1] S. M. Sze, *Physics of Semiconductor Devices*, 2nd edition, John Wiley & Sons, Inc., USA, 1981.
- [2] H. Chan and T. Shieh, "A Three-Dimensional Semiconductor Device Simulator for GaAs/AlGaAs Heterojunction Bipolar Transistor Analysis", *IEEE Transactions on Electron Devices*, **38** (11), pp. 2427-2432, 1991.
- [3] K Yang, J. R. East, and G. I. Haddad, "Numerical Modeling of Abrupt Heterojunctions using Thermionic-field Emission Boundary Condition", *Solid-State Electronics*, **36** (3), pp. 321-330, 1993.
- [4] A. M. Cruz Serra and H. Abreu Santos, "A one-dimensional, self-consistent numerical solution of Schrodinger and Poisson equations", *Journal of Applied Physics*, **70**, pp. 2734-2738, 1991.

Output จากโครงการวิจัยที่ได้รับทุนจาก สกว.

- ผลงานตีพิมพ์ในวารสารวิชาการนานาชาติ

P. Aiyarak, A. J. Vickers and P. Boonlua, "A Computational Search for the Optimized Structure of a Novel Light Emitter", Journal of Physics D: Applied Physics. (อยู่ในช่วงปรับปรุง Manuscript กับผู้เขียนในต่างประเทศเพื่อเตรียมส่ง)

- การนำผลงานไปใช้ประโยชน์

เชิงพาณิชย์

โครงการนี้มีศักยภาพมากในเชิงพาณิชย์หากโปรแกรมได้รับการปรับปรุงให้ดีขึ้นในการใช้งานและสามารถรองรับชนิดของสารกึ่งตัวนำได้มากขึ้น รวมทั้งสามารถเพิ่มเติม (Upgrade) ข้อมูลชนิดของสารกึ่งตัวนำ นอกจากนี้หากสามารถแก้ปัญหาการลู่เข้า (convergence) ของผลเฉลยในทุกกรณีในการแก้สมการแบบ self-consistence ของสมการ drift-diffusion และ Schrödinger ซึ่งหากได้รับการปรับปรุงดังกล่าวแล้วจะทำให้สามารถทำนายได้ในอุตสาหกรรมที่เกี่ยวข้องกับการออกแบบสิ่งประดิษฐ์สารกึ่งตัวนำ

เชิงสาธารณะ

ในภาพรวมสามารถนำโปรแกรมที่ได้ไปออกแบบสิ่งประดิษฐ์สารกึ่งตัวนำใหม่ๆ ซึ่งทำให้เกิดความร่วมมือกันเป็นเครือข่ายได้ ทั้งในองค์กรเอกชน และองค์กรของรัฐ

เชิงวิชาการ

ทำให้การเรียนการสอนในเรื่องสารกึ่งตัวนำ และสิ่งประดิษฐ์สารกึ่งตัวนำทำได้ง่ายขึ้น โดยนักศึกษาสามารถใช้โปรแกรมเพื่อศึกษาชนิดของสิ่งประดิษฐ์สารกึ่งตัวนำต่างๆ ได้ และสามารถนำมาใช้เป็นเครื่องมือในการวิจัยเพื่อออกแบบสิ่งประดิษฐ์สารกึ่งตัวนำต่างๆ ได้

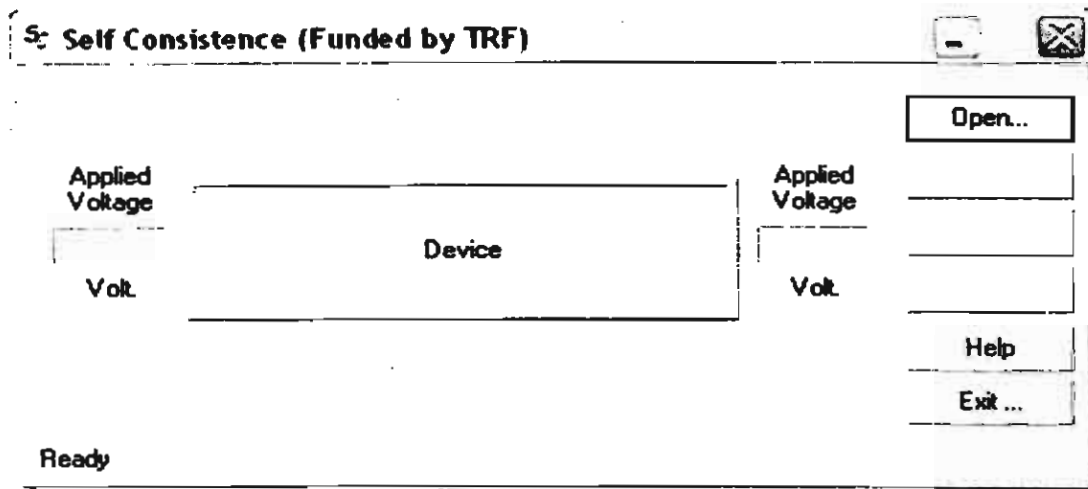
- อื่นๆ

งานประชุมวิชาการ วิทยาศาสตร์และเทคโนโลยีแห่งประเทศไทย (วทท.) ครั้งที่ 28 พ.ศ. 2545 ณ กรุงเทพมหานคร

ภาคผนวก ก.

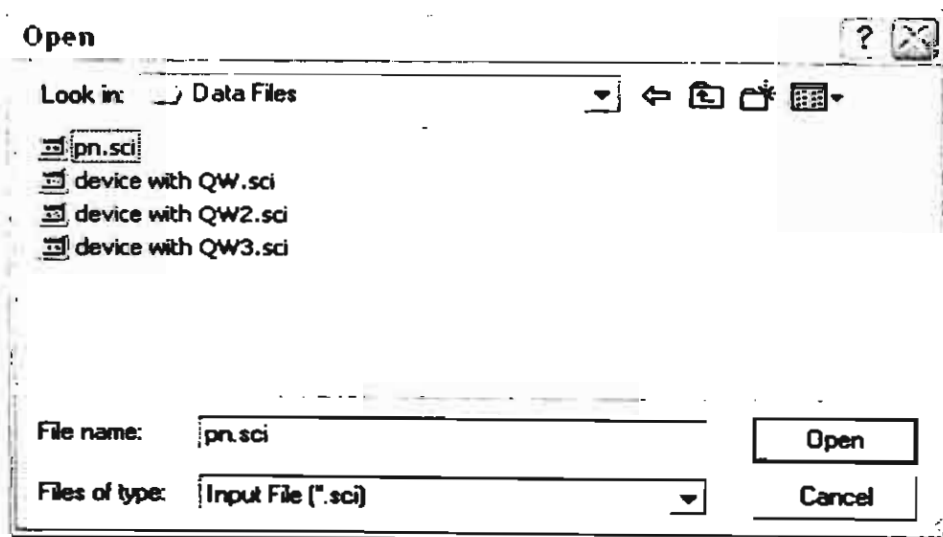
การใช้งานโปรแกรมพร้อม Screen Shots ตัวอย่าง

ตัวอย่างของ Screen Shots ซึ่งเป็นส่วนของการจำลองแบบซึ่งทำงานบนเครื่องคอมพิวเตอร์ส่วนบุคคลทั่วไปซึ่งโปรแกรมจะปรากฏขึ้นมาดังภาพที่ ก(1) ผู้ใช้จะต้องทำการเรียกแฟ้มที่เก็บโครงสร้างของสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งมีรูปแบบดังภาคผนวก ข.



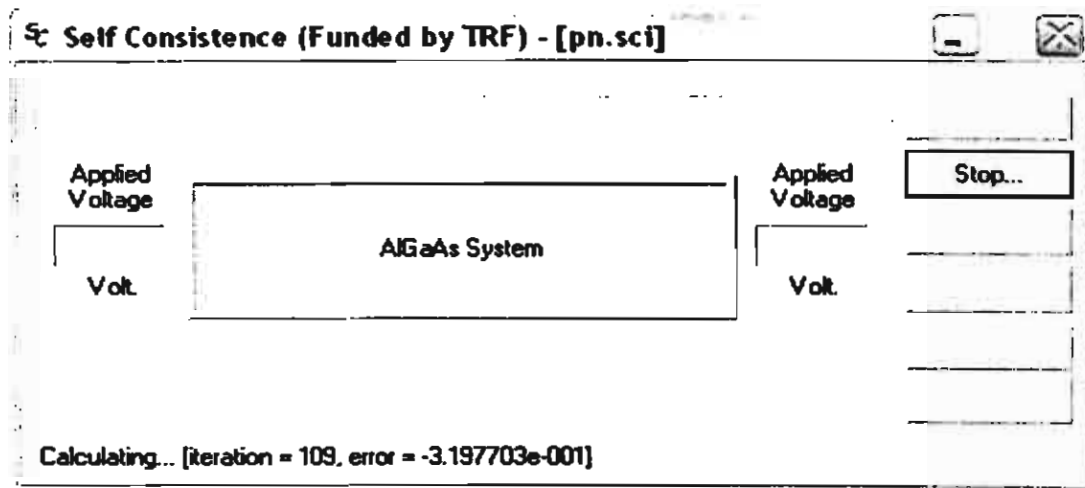
ภาพ ก(1): ภาพแสดง Screen Shot ของโปรแกรม

หลังจากกดปุ่ม Open... แล้วจะปรากฏ dialog box สำหรับเลือกแฟ้มเก็บโครงสร้างของสิ่งประดิษฐ์สารกึ่งตัวนำ ดังในภาพที่ ก(2)



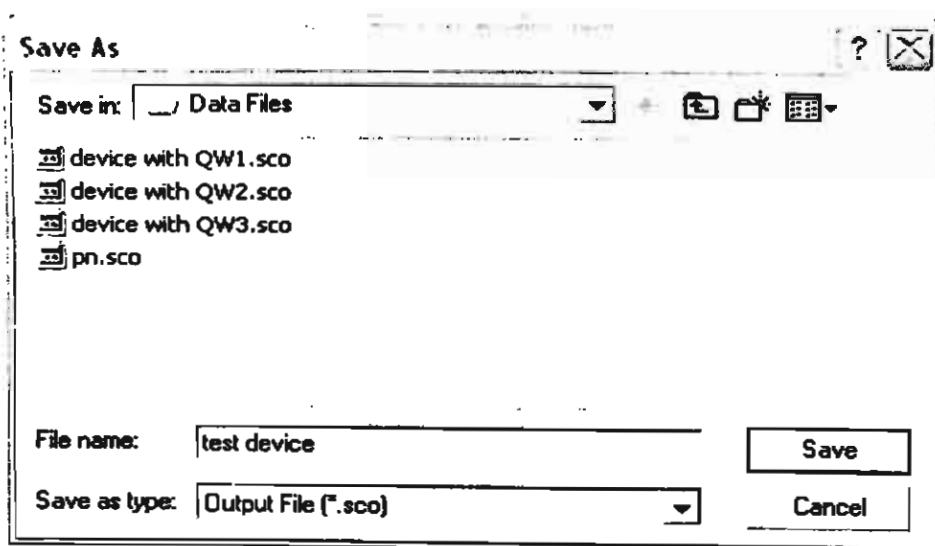
ภาพ ก(2): ภาพแสดง Screen Shot ของ Open Dialog

หลังจากเลือกเพิ่มเก็บโครงสร้างของสิ่งประดิษฐ์สารกึ่งตัวนำแล้ว ให้กดปุ่ม Calculate ซึ่งจะทำให้ตัวโปรแกรมเริ่มการทำงานดังจะเห็นได้จาก Status Bar ด้านล่างแสดงจำนวน Iteration หรือจำนวนรวมในการคำนวณ และค่า Error ซึ่งเป็นค่าความแตกต่างระหว่างผลเฉลยใน Iteration ก่อนหน้า ดังแสดงในภาพ ก(3) หากค่า Error ที่ได้มีค่าต่ำแสดงให้เห็นว่าผลเฉลยที่ได้เริ่มมีค่าคงที่ ซึ่งค่า Error ที่รับได้ควรมีค่าต่ำกว่า 1×10^{-5} และหากค่าของผลเฉลยถึงจุดที่ไม่เปลี่ยนแปลงมากผู้ใช้สามารถกดปุ่ม Stop... เพื่อหยุดการทำงาน



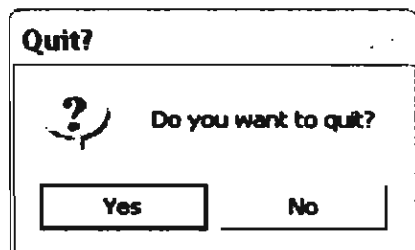
ภาพ ก(3): ภาพแสดง Screen Shot ของ Open Dialog

เมื่อโปรแกรมหยุดการทำงานหลังผู้ใช้กดปุ่ม Stop... ผู้ใช้สามารถบันทึกผลเฉลยที่ได้ลงในแฟ้มโดยการกดปุ่ม Save... ซึ่งโปรแกรมจะแสดง Dialog Save เพื่อผู้ใช้สามารถบันทึกผลเฉลยที่ต้องการลงในแฟ้มในชื่อที่ผู้ต้องการดังภาพ ก(4)



ภาพ ก(4): ภาพแสดง Screen Shot ของ Save Dialog

หากผู้ใช้ต้องการจบการทำงานสามารถทำได้โดยการกดปุ่ม Exit... ซึ่งจะได้ Dialog ยืนยันการจบการทำงานของโปรแกรม ดังภาพ ก(5) หากต้องการจบการทำงานให้กดปุ่ม Yes หรือหากต้องการกลับไปทำงานต่อให้กด No โปรแกรมจะกลับไปยัง Dialog หลักซึ่งผู้ใช้สามารถทำการคำนวณหาผลเฉลยต่อไปได้



ภาพ ก(5): ภาพแสดง Screen Shot ของการยืนยันการจบการทำงาน

ภาคผนวก ข.

โครงสร้างของแฟ้มข้อมูล Input และ Output

Input File

แฟ้มที่เป็น Input File จะเป็นแฟ้มที่มีนามสกุล sci ซึ่งรูปแบบของแฟ้มประกอบด้วยสองบรรทัดแรกคือ

```
#Do not change this line or the program won't work.
```

```
#position x_right(cm) x_left(cm) temperature(K) doping(cm-3) QW(y/n) material composite1 composite2
```

หลังจากนั้นเป็นโครงสร้างของสิ่งประดิษฐ์สารกึ่งตัวนำซึ่งประกอบด้วยข้อมูลตามลำดับดังนี้

- ตำแหน่งของสาร ซึ่งแบ่งเป็น Mesh Point
- ระยะห่างระหว่าง Mesh Point เมื่อเทียบกับ Mesh Point ถัดไป (หน่วย cm)
- ระยะห่างระหว่าง Mesh Point เมื่อเทียบกับ Mesh Point ก่อนหน้าไป (หน่วย cm)
- อุณหภูมิ (หน่วย K)
- Doping (หน่วย cm^{-3}) ซึ่งหากเป็นชนิด n จะมีค่าเป็นบวก และลบหากเป็นชนิด p
- QW (หากจุดดังกล่าวเป็น Quantum Well ให้ใส่ y)
- ชนิดของสารกึ่งตัวนำ (โปรแกรมรองรับเพียง AlGaAs ในขณะนี้แต่สามารถเพิ่มเติมได้ในอนาคต)
- ค่าคงที่อื่นๆ ตัวที่ 1 ซึ่งในกรณีนี้คือค่า x ใน $\text{Al}_x\text{Ga}_{1-x}\text{As}$
- ค่าคงที่อื่นๆ ตัวที่ 2 ซึ่งเพื่อไว้ในอนาคต (หากไม่มีให้ใส่ 0.0 ห้ามทิ้งว่างไว้)

ตัวอย่าง Input File

```
#Do not change this line or the program won't work.
```

```
#position x_right(cm) x_left(cm) temperature(K) doping(cm-3) QW(y/n) material composite1 composite2
```

```
1 5e-7 5e-7 300 1e18 n AlGaAs 0.0 0.0
2 5e-7 5e-7 300 1e18 n AlGaAs 0.0 0.0
3 5e-7 5e-7 300 1e18 n AlGaAs 0.0 0.0
4 5e-7 5e-7 300 1e18 n AlGaAs 0.0 0.0
5 5e-7 5e-7 300 1e18 n AlGaAs 0.0 0.0
6 5e-7 5e-7 300 1e18 n AlGaAs 0.0 0.0
7 5e-7 5e-7 300 1e18 n AlGaAs 0.0 0.0
8 5e-7 5e-7 300 1e18 n AlGaAs 0.0 0.0
9 5e-7 5e-7 300 -1e18 n AlGaAs 0.0 0.0
10 5e-7 5e-7 300 -1e18 n AlGaAs 0.0 0.0
11 5e-7 5e-7 300 -1e18 n AlGaAs 0.0 0.0
12 5e-7 5e-7 300 -1e18 n AlGaAs 0.0 0.0
13 5e-7 5e-7 300 -1e18 n AlGaAs 0.0 0.0
14 5e-7 5e-7 300 -1e18 n AlGaAs 0.0 0.0
15 5e-7 5e-7 300 -1e18 n AlGaAs 0.0 0.0
16 5e-7 5e-7 300 -1e18 n AlGaAs 0.0 0.0
```

Output File

ผลเฉลยที่ได้สามารถบันทึกลงในแฟ้มนามสกุล sco ซึ่งประกอบด้วยองค์ประกอบดังนี้

- ตำแหน่งของ Mesh Point
- Conduction Energy (EC)
- Valence Energy (EV)
- ความหนาแน่นของอิเล็กตรอน
- ความหนาแน่นของโฮล
- ระยะห่างเมื่อเทียบกับ Mesh Point ถัดไป
- ศักย์ ณ Mesh Point นั้นๆ

ตัวอย่าง Output File

#position	Ec	Ev	n	p	x_right	potential
0	-4.772553	-6.194953	1.00E+18	3.39E-06	5.00E-09	7.03E-01
1	-4.760746	-6.183146	6.35E+17	5.33E-06	5.00E-09	6.91E-01
2	-4.748921	-6.171321	4.03E+17	8.40E-06	5.00E-09	6.79E-01
3	-4.737069	-6.159469	2.55E+17	1.33E-05	5.00E-09	6.67E-01
4	-4.72518	-6.14758	1.62E+17	2.09E-05	5.00E-09	6.55E-01
5	-4.713247	-6.135647	1.02E+17	3.31E-05	5.00E-09	6.43E-01
6	-4.701262	-6.123662	6.44E+16	5.25E-05	5.00E-09	6.31E-01
7	-4.689216	-6.111616	4.05E+16	8.35E-05	5.00E-09	6.19E-01
8	-4.677102	-6.099502	2.54E+16	1.33E-04	5.00E-09	6.07E-01
9	-4.664912	-6.087312	1.59E+16	2.13E-04	5.00E-09	5.95E-01
10	-4.652641	-6.075041	9.93E+15	3.41E-04	5.00E-09	5.83E-01
11	-4.640282	-6.062682	6.17E+15	5.48E-04	5.00E-09	5.70E-01
12	-4.627828	-6.050228	3.82E+15	8.85E-04	5.00E-09	5.58E-01
13	-4.615274	-6.037674	2.36E+15	1.43E-03	5.00E-09	5.45E-01
14	-4.602614	-6.025014	1.45E+15	2.33E-03	5.00E-09	5.33E-01
15	-4.589844	-6.012244	8.87E+14	3.82E-03	5.00E-09	5.20E-01
16	-4.576958	-5.999358	5.41E+14	6.26E-03	5.00E-09	5.07E-01
17	-4.563954	-5.986354	3.28E+14	1.03E-02	5.00E-09	4.94E-01
18	-4.550828	-5.973228	1.98E+14	1.71E-02	5.00E-09	4.81E-01
19	-4.537576	-5.959976	1.19E+14	2.85E-02	5.00E-09	4.68E-01
20	-4.524196	-5.946596	7.10E+13	4.77E-02	5.00E-09	4.54E-01

ภาคผนวก ค.

Class และ Main Program สำหรับโปรแกรมพ UNIX

● Class header (material.h)

```
// header file for the object "material" //

class material {

protected :                // list of protected variables

    double phi;              // potential
    double field;            // electric field
    double U;                // special variable for electrons
    double V;                // special variable for holes
    double n;                // electron concentration
    double p;                // hole concentration

    double density;          // density of the material
    double phi0;             // initial potential
    double phi_applied;      // applied potential
    double ni;               // ni of the material
    double q;                // electron charge
    double eps;              // material permittivity
    double Eg;               // material bandgap
    double chi;              // electron affinity of the material
    double VT;               // kT/q
    double Temperature;      //
    double x_right;          // hi,j
    double x_left;           // hi-1,j
    double y_top;            // ki,j
    double y_bottom;         // ki,j-1
    double Nc;               // effective density of states in conduction band
    double Nv;               // effective density of states in valence band
    double Tp;               // hole lifetime
    double Tn;               // electron lifetime
    double Dn;               // electron diffusion coefficient
    double Dp;               // hole diffusion coefficient

public :                    // list of public functions

    material();              // default constructor
    material(double density_p, double phi_applied_p = 0,
              double ni_p = 5915256.5520340,
              double eps_p = 1.16644e-12, double Eg_p = 0,
              double chi_p = 0, double Nc_p = 1, double Nv_p = 1,
              double Temperature_p = 300, double q_p = 1.6e-19,
              double VT_p = 0.026, double x_right_p = 5e-7,
              double x_left_p = 5e-7, double y_top_p = 5e-7,
              double y_bottom_p = 5e-7, double Tp_p = 1e-8,
              double Tn_p = 1e-8, double Dp_p = 10.4,
              double Dn_p = 221); // main constructor
    ~material() {}           // destructor

    double potential() { return phi; } // return potential
    double u() { return U; }           // return U
    double v() { return V; }           // return V
    double EG() { return Eg; }         // return Eg
    double CHI() { return chi; }       // return chi
    double ret_n() { return n; }       // return n
    double ret_p() { return p; }       // return p
    double ret_Ni() { return ni; }     // return ni
    double ret_VT() { return VT; }     // return VT
    double ret_Dn() { return Dn; }     // return Dn
    double ret_Dp() { return Dp; }     // return Dp
    double ret_Field() { return field; } // return field
    double ret_Fermi_leveln(double px); // return electron quasi-fermi
    double ret_Fermi_levelp(double px); // return electron quasi-fermi

    // calculate phi from the neighbour's objects
    double cal_phi(material &pi, material &mi, material &jp1,
                   material &jm1, double error_limit = 1e-7,
                   int m = 1000);

```

```

// calculate U from the neighbour's objects
material cal_U(material &ip1, material &im1, material &jp1,
               material &jm1);

// calculate V from the neighbour's objects
material cal_V(material &ip1, material &im1, material &jp1,
               material &jm1);

// calculate the electron density in the quantum well
material cal_qwn(double Efermi, double size, double perc);

// calculate the hole density in the quantum well
material cal_qwp(double Efermi, double size, double perc);

// calculate field from the neighbour's objects
material cal_field(material &ip1, material &im1, material &jp1,
                  material &jm1);

material setDensity(double denst);           // set material density
material setAppliedVolt(double phi_appl);    // set the applied voltage
material set_Diffusivity();                  // set Dn an Dp
material set_time(double tau);                // set lifetime
material set_step(double size1, double size2); // set step sizes
material operator-(const material &input);    // declare equal sign
friend ostream &operator<<(ostream &stream,
                           const material &output); // declare output
};

```

● Class implementation (material.cpp)

```

#include <iostream.h>
#include <math.h>
#include <std.lib.h>
#include "material.h"

material::material() { // method for default constructor

    phi = 0;
    field = 0;
    phi0 = 0;
    phi_applied = 0;
    V = 0;
    E = 0;
    p = 0;
    density = 0;
    ni = 0;
    T = 0;
    eps = 0;
    Eg = 0;
    chi = 0;
    VT = 0;
    Temperature = 0;
    x_right = 0;
    x_left = 0;
    y_top = 0;
    y_bottom = 0;
    Nv = 0;
    Nd = 0;
    qn = 0;
    qh = 0;
    q = 0;
    Tp = 0;
    Tn = 0;

}

// method for constructor
material::material(double density_p, double phi_applied_p, double ni_p,
                  double eps_p, double Eg_p, double chi_p, double Nd_p,
                  double Nv_p, double Temperature_p, double q_p, double VT_p,
                  double x_right_p, double x_left_p, double y_top_p,
                  double y_bottom_p, double Tp_p, double Tn_p, double Dp_p,
                  double Dn_p) {

    double temp;

```

```

density      = density_p;
phi_applied  = phi_applied_p;
ni           = ni_p;
q            = q_p;
eps          = eps_p;
Eg           = Eg_p;
chi          = chi_p;
VT           = VT_p;
Temperature  = Temperature_p;
x_right      = x_right_p;
x_left       = x_left_p;
y_top        = y_top_p;
y_bottom     = y_bottom_p;
Nc           = Nc_p;
Nv           = Nv_p;
Tp           = Tp_p;
Tn           = Tn_p;
Dn           = Dn_p;
Dp           = Dp_p;

if (density < 0) {
    temp = fabs(density) / (2 * ni);
    phi0 = -log(temp + sqrt((temp*temp) + 1));
}
else {
    temp = density / (2 * ni);
    phi0 = log(temp + sqrt((temp*temp) + 1));
}

phi = phi_applied + phi0*VT;
U   = exp(-phi_applied/VT);
V   = exp(phi_applied/VT);
n = ni * Nc * U * exp(phi/VT)*exp(chi/q/VT);
p = ni * Nv * V * exp(-phi/VT)*exp(-(chi+Eg)/q/VT);
}

// method for calculating the potential
double material::cal_phi(material& ipl, material& iml, material& jpl,
                        material& jml, double error_limit, int m) {

    double y, dy, dxeps, dyeps;
    double aij, aiplj, aimlj, aijpl, aijml;
    double F, f;
    double G1, G2;

    if (m <= 0) {
        cout << "It's not funny" << m << ", set 'm' again.\n";
        exit(1);
    }

    dxeps = ipl.eps/x_right/2-iml.eps/x_left/2-eps*(1/x_right-1/x_left)/2;
    dyeps = jpl.eps/y_top/2-jml.eps/y_bottom/2-eps*(1/y_top-1/y_bottom)/2;
    aiplj = eps*2. / (x_right + x_left) / x_right+dxeps/x_right/2;
    aimlj = eps*2. / (x_right + x_left) / x_left-dxeps/x_left/2;
    aijpl = eps*2. / (y_top + y_bottom) / y_top+dyeps/y_top/2;
    aijml = eps*2. / (y_top + y_bottom) / y_bottom-dyeps/y_bottom/2;
    aij = aiplj + aimlj + aijpl + aijml;

    f = -q*density;
    F = aiplj*ipl.phi + aimlj*iml.phi + aijpl*jpl.phi + aijml*jml.phi - f;
    G1 = q*ni*(Nc*U*exp(phi/VT)*exp(chi/VT/q)
            -Nv*V*exp(-phi/VT)*exp(-(chi+Eg)/q/VT));
    y = aij*phi + G1 - F;

    while ((fabs(y) > error_limit) && (m > 0)) {
        m--;
        G2 = q * ni * (Nc*U/VT*exp(phi/VT)*exp(chi/VT)
                    + Nv*V/VT*exp(-phi/VT)*exp(-(chi+Eg)/q/VT));
        dy = aij + G2;
        phi -= y/dy;
        G1 = q*ni*(Nc*U*exp(phi/VT)*exp(chi/VT/q)
                - Nv*V*exp(-phi/VT)*exp(-(chi+Eg)/q/VT));
        y = aij*phi + G1 - F;
    }

    if (m < 0) {

```

```

    cout << "dy = " << dy << "\n";
    cout << "1/dy = " << 1/dy << "\n";
    cout << "Function is not convergent, exit program.\n";
    exit(1);
}

n = ni*Nc * U * exp(phi/VT)*exp(chi/q/VT);
p = ni* Nv * V * exp(-phi/VT)*exp(-(chi+Eg)/q/VT);

return phi;
}

// method for calculating U
material material::cal_U(material& ipl, material& iml, material& jpl,
                        material& jml) {

    double aij, aiplj, aimlj, aijpl, aijml;
    double cnhp, cnhm, cnkp, cnkm, bnhp, enhp, xnhp, bnhm, xnhm;
    double bnkp, xnkp, bnkm, xnkml;
    double F, f, G, r;

    cnhp = 2*Dn*Nc / (x_right + x_left) / x_right;
    xnhp = (phi - ipl.phi) / VT + (chi - ipl.chi) / q / VT;
    bnhp = fabs(xnhp) < 1e-10 ? 1-0.5*xnhp : xnhp / (exp(xnhp) - 1);
    enhp = exp((phi+chi/q)/VT);
    aiplj = cnhp * bnhp;

    cnhm = 2*Dn*Nc / (x_right + x_left) / x_left;
    xnhm = (phi - iml.phi) / VT + (chi - iml.chi) / q / VT;
    bnhm = fabs(xnhm) < 1e-10 ? 1-0.5*xnhm : xnhm / (exp(xnhm) - 1);
    aimlj = cnhm * bnhm;

    cnkp = 2*Dn*Nc / (y_top + y_bottom) / y_top;
    xnkp = (phi - jpl.phi) / VT + (chi - jpl.chi) / q / VT;
    bnkp = fabs(xnkp) < 1e-10 ? 1-0.5*xnkp : xnkp / (exp(xnkp) - 1);
    aijpl = cnkp * bnkp;

    cnkm = 2*Dn*Nc / (y_top + y_bottom) / y_bottom;
    xnkml = (phi - jml.phi) / VT + (chi - jml.chi) / q / VT;
    bnkm = fabs(xnkml) < 1e-10 ? 1-0.5*xnkml : xnkml / (exp(xnkml) - 1);
    aijml = cnkm * bnkm;

    aij = aiplj + aimlj + aijpl + aijml;

    r = 1.0 / (Tn*ni*( Nv*V*exp(-phi/VT)*exp(-chi/q/VT)+1)
              + Tp*ni*(Nc*U*exp(phi/VT)*exp(chi/VT/q)+1));
    G = ni * V * r/enhp;
    f = -ni * r/enhp;
    F = aiplj*ipl.U + aimlj*iml.U + aijpl*jpl.U + aijml*jml.U - f;

    U = F / (aij+G);

    n = ni* Nc * U * exp(phi/VT)*exp(chi/q/VT);

    return U;
}

// method for calculating the V
material material::cal_V(material& ipl, material& iml, material& jpl,
                        material& jml) {

    double aij, aiplj, aimlj, aijpl, aijml;
    double cphp, cphm, cpkp, cpkm, bphp, ephp, xphp, bphm, xphm;
    double bpkp, xpkp, bpkml, xpkml;
    double F, f, G, r;

    cphp = 2*Dp*Nv / (x_right + x_left) / x_right;
    xphp = (ipl.phi - phi) / VT + (ipl.chi - chi) / q / VT + (ipl.Eg - Eg) / q / VT;
    bphp = fabs(xphp) < 1e-10 ? 1+0.5*xphp : xphp / (exp(xphp) - 1);
    ephp = exp(-(phi+chi/q+Eg/q)/VT);
    aiplj = cphp * bphp;

    cphm = 2*Dp*Nv / (x_right + x_left) / x_left;
    xphm = (iml.phi - phi) / VT + (iml.chi - chi) / q / VT + (iml.Eg - Eg) / q / VT;
    bphm = fabs(xphm) < 1e-10 ? 1+0.5*xphm : xphm / (exp(xphm) - 1);
    aimlj = cphm * bphm;

```

```

cpkp = 2*Dp*Nv / (y_top + y_bottom) / y_top;
xpkp = (jpl.phi - phi) / VT + (jpl.chi - chi) / q / VT + (jpl.Eg - Eg) / q / VT;
bpkp = fabs(xpkp) < 1e-10 ? 1 + 0.5*xpkp : xpkp / (exp(xpkp) - 1);
aijpl = cpkp * bpkp;

cpkm = 2*Dp*Nv / (y_top + y_bottom) / y_bottom;
xpkm = (jml.phi - phi) / VT + (jml.chi - chi) / q / VT + (jml.Eg - Eg) / q / VT;
bpkm = fabs(xpkm) < 1e-10 ? 1 + 0.5*xpkm : xpkm / (exp(xpkm) - 1);
aijml = cpkm * bpkm;

aij = aiplj + aimlj + aijpl + aijml;

r = 1. / (Tn*ni*(Nv*V*exp(-phi/VT)*exp(-(chi+Eg)/q/VT)+1)
          + Tp*ni*(Nc*U*exp(phi/VT)*exp(chi/q/VT)+1));
G = ni * U * r / eph;
f = -ni * r / eph;
F = aiplj*ipl.V + aimlj*iml.V + aijpl*jpl.V + aijml*jml.V - f;

V = F / (aij+G);

p = ni * Nv * V * exp(-phi/VT)*exp(-(chi+Eg)/q/VT);

return V;
}

// method to calculate the electron quasi-Fermi level
double material::ret_Fermi_leveln (double px) {

    double Efi, Ef;
    double me = (0.067 + 0.083*px) * 0.91e-30;
    double mlh = 0.087 + 0.063*px;
    double mhh = 0.62 + 0.14*px;
    double templ = sqrt(mlh*mlh*mlh) + sqrt(mhh*mhh*mhh);
    double mh = cbrt(templ*templ) * 0.91e-30;

    const double CHI = 4.07*q;
    const double EGAP = 1.4224*q;

    // mh=0.45*9.108e-31;
    // me=0.067*9.108e-31;

    VT=0.0259;
    q = 1.6e-19;
    Efi=-phi*q-(chi-CHI)-(Eg+EGAP)/2+3/4*VT*q*log(mh/me);
    Ef=Efi-q*(phi+chi/q+VT*log(Nc)+VT*log(U));

    return Ef;
}

// method to calculate the hole quasi-Fermi level
double material::ret_Fermi_levelp (double px) {

    double Efi, Ef;

    double me = (0.067 + 0.083*px) * 0.91e-30;
    double mlh = 0.087 + 0.063*px;
    double mhh = 0.62 + 0.14*px;
    double templ = sqrt(mlh*mlh*mlh) + sqrt(mhh*mhh*mhh);
    double mh = cbrt(templ*templ) * 0.91e-30;

    const double CHI = 4.07*q;
    const double EGAP = 1.4224*q;

    // mh=0.45*9.108e-31;
    // me=0.067*9.108e-31;

    VT=0.0259;
    q = 1.6e-19;
    Efi=-phi*q-(chi-CHI)-(Eg+EGAP)/2+3/4*VT*q*log(mh/me);
    Ef=Efi+q*(phi+chi/q+Eg/q-VT*log(Nv)-VT*log(V));

    return Ef;
}

// method to calculate the electron density in the quantum well

```

```

material material::cal_qwn (double Efermi,double size,double perc) {

    int num,i;
    double tmp = 0;
    VT=0.0259;
    q = 1.6e-19;
    double V0, gam, me, htagl, mh, lf, rg, a_temp, y;
    double Energ[30];
    double a[30];

    const double CHI = 4.07*q;
    const double EGAP = 1.4224*q;

    mh=0.45*9.108e-31;
    me=0.067*9.108e-31;
    htagl=6.625e-34/(2*3.14159);
    V0=0.7482*(perc)*q;
    gam=sqrt(2*(me/htagl)*V0*size*size/htagl);

    num=0;

    do {
        lf=0;
        rg=1;
        num++;

        do {
            a_temp=(lf+rg)/2;
            y=gam*a_temp-((num-1)*3.14159+2*acos(a_temp));

            if (y<0)
                lf=a_temp;
            else
                rg=a_temp;
        } while((fabs(y)>1e-5)&&((1-lf)>1e-8));

        Energ[num]=V0*a_temp*a_temp;
        a[num]=a_temp;

    } while(((1-lf)>1e-8) && (num<30));

    n=0;

    for(i=1;i<num;i++) {
        tmp = log(1+exp((Efermi+phi*q+(chi+CHI)-Energ[i])/(VT*q)));
        if (tmp>=0)
            tmp = exp((Efermi+phi*q+(chi+CHI)-Energ[i])/(VT*q));
        n += 7.232e+15*tmp;
    }

    U = exp(-phi/VT)*exp(-chi/q/VT)* n /(ni*Nc);

    return *this;
}

// method to calculate the hole density in the quantum well
material material::cal_qwp (double Efermi,double size,double perc) {

    int num, i;
    double tmp;

    VT = 0.0259;
    q = 1.6e-19;

    double V0, gam, me, htagl, mh, lf, rg, a_temp, y;
    double Energ[30];
    double a[30];

    const double CHI = 4.07*q;
    const double EGAP = 1.4224*q;

    mh = 0.45*9.108e-31;
    me = 0.067*9.108e-31;
    htagl = 6.625e-34/(2*3.14159);
    V0 = (1.247-0.7482)*(perc)*q;
    gam = sqrt(2*mh*V0*size*size/(htagl*htagl));
    num = 0;

```



```

do {
    lf=0;
    rg=1;
    num++;

    do {
        a_temp=(lf+rg)/2;
        y=gam*a_temp-((num-1)*3.14159+2*acos(a_temp));

        if (y<0)
            lf=a_temp;
        else
            rg=a_temp;

    } while((fabs(y)>1e-5)&&((1-lf)>1e-8));

    Energ[num]=V0*a_temp*a_temp;
    a[num]=a_temp;

} while(((1-lf)>1e-8)&&(num<30));

p = 1.0e-10;

for(i=1;i<num;i++) {
    tmp=log(1+exp((-phi*q-(chi+CHI)-(Eg+EGAP)-Energ[i]-Efermi)/(VT*q)));
    if (tmp==0)
        tmp = exp((-phi*q-(chi+CHI)-(Eg+EGAP)-Energ[i]-Efermi)/(VT*q));

    p += 4.857e+16*tmp;
}

V = p/(ni*Nv) * exp(phi/VT) * exp((chi+Eg)/q/VT);

return *this;
}

// method to calculate the field
material material::cal_field(material& ipl, material& iml, material& jpl,
                             material& jml) {

    double dfx,dfy;
    dfx=-(ipl.phi-phi)/x_right/2-(phi-iml.phi)/x_left/2;
    dfy=-(jpl.phi-phi)/y_top/2-(phi-jml.phi)/y_bottom/2;
    field=sqrt(dfx*dfx + dfy*dfy);
    return *this;
}

// method to set the density of the material
material material::setDensity( double denst) {

    double temp;

    density = denst;

    if (density < 0) {
        temp = fabs(density) / (2 * ni);
        phi0 = -log(temp + sqrt((temp*temp) + 1));
    }
    else {
        temp = density / (2 * ni);
        phi0 = log(temp + sqrt((temp*temp) + 1));
    }

    phi = phi_applied + phi0*VT;
    n = ni*Nc * U * exp(phi/VT) * exp(chi/VT/q);
    p = ni*Nv * V * exp(-phi/VT) * exp(-(chi+Eg)/VT/q);

    return *this;
}

// method to set the applied voltage
material material::setAppliedVolt(double phi_appl) {

    phi_applied = phi_appl;
    phi = phi_applied + phi0*VT;
    U = exp(-phi_applied/VT);

```

```

V          = exp(phi_applied/VT);

n = ni* Nc * U * exp(phi/VT) * exp(chi/VT/q);
p = ni* Nv * V * exp(-phi/VT) * exp(-(chi +Eg)/VT/q);

return *this;
}

// method to set the lifetime
material material::set_time( double tau) {

    Tp=tau;
    Tn=tau;
    return *this;
}

// method to set the diffusion coefficients
material material::set_Diffusivity() {

    double mobn0,mobp0;
    mobn0 = (7200/(pow(1+5.51e-17*fabs(density),0.233)))
            *pow(300/Temperature, 2.3);
    mobp0 = (380/(pow(1+3.17e-17*fabs(density),0.266)))
            *pow(300/Temperature, 2.7);
    Dn=VT*(mobn0+7.7e06*pow(field,3)/2.56e14)/(1+pow(field/4e03,4));
    Dp=VT*mobp0/(1+mobp0*field/1.5e07);

    return *this;
}

// method to set the steps (h, k)
material material::set_step( double size1, double size2) {

    x_left = size1;
    x_right = size1;
    y_top = size2;
    y_bottom = size1;

    return *this;
}

// declaration for the equal sign
material material::operator=(const material &input) {

    phi          = input.phi;
    U            = input.U;
    V            = input.V;
    phi0         = input.phi0;
    phi_applied  = input.phi_applied;
    density      = input.density;
    ni           = input.ni;
    n            = input.n;
    p            = input.p;
    q            = input.q;
    eps          = input.eps;
    Eg           = input.Eg;
    chi          = input.chi;
    VT           = input.VT;
    Temperature  = input.Temperature;
    x_right      = input.x_right;
    x_left       = input.x_left;
    y_top        = input.y_top;
    y_bottom     = input.y_bottom;
    Tp           = input.Tp;
    Tn           = input.Tn;
    Dp           = input.Dp;
    Dn           = input.Dn;
    Nc           = input.Nc;
    Nv           = input.Nv;

    return *this;
}

// declaration for the object when used with the cout function.
ostream& operator<<(ostream &stream, const material& output) {

    stream << "Potential = " << output.phi << "\n";

```

```
stream << "U = " << output.U << "\n";
stream << "V = " << output.V << "\n";
```

```
return stream;
}
```

● Main Program

```
#include <iostream.h>
#include <fstream.h>
#include <math.h>
#include <stdlib.h>
#include "material.h"

double eps_AlGaAs(double x=0);          // return eps of AlGaAs
double ni_AlGaAs(double x=0);           // return ni of AlGaAs
double chi_AlGaAs(double x=0);          // return chi of AlGaAs
double Nc_AlGaAs(double x=0);           // return Nc of AlGaAs
double Nv_AlGaAs(double x=0);           // return Nv of AlGaAs
double Eg_AlGaAs(double x=0);           // return Eg of AlGaAs

void main(void) {

    /*** declarations ***/
    const int max_i = 122;
    const int max_j = 3;
    const double density_of_n = 5e16;
    const double density_of_i = 1e15;
    const double density_of_p_contact = -1e17;
    const double density_of_barrier = 5e17;
    const double density_of_QW = 5e14;
    const double q = 1.6e-19;
    const double eps_GaAs = 13.18 * 8.85e-14;
    const double CHI = 4.07*q;
    const double EGAP = 1.4224*q;
    const double NCR = 2.5e19*0.067;
    const double NVR = 2.5e19*0.48;
    const double ni_GaAs = 5915256.5520340;
    const double VT = 0.026;

    double Ni_AlGaAs;
    int i,j;
    double vectx[max_i+2];
    double x;
    double vn,vp;
    double vj;
    material mat[max_i+2][max_j+2];
    ofstream dmddata;

    // initialise the GaAs
    material *temp_GaAs = new material(0,0,ni_GaAs,eps_GaAs,0,0,1,1,300);

    vn = 0.0;          // applied voltage at n-doped region
    vp = 0.0;          // applied voltage at p-doped region

    for (i=1; i<=max_i; i++)          // looping for the whole device
        for (j=1; j<=max_j; j++) {

            if (i>=1 && i<=40) { // set the dimension and parameters of n region

                x = 0.1;
                vectx[i]=x;          // Al composition in AlGaAs of n region

                // calculate the ni
                Ni_AlGaAs = sqrt(Nc_AlGaAs(x)*Nv_AlGaAs(x))
                    * exp(-(Eg_AlGaAs(x)/2)/q)/VT);

                // initialise the dummy
                material *temp_AlGaAs = new material(0,0,Ni_AlGaAs,
                    eps_AlGaAs(x),
                    Eg_AlGaAs(x)-EGAP,
                    chi_AlGaAs(x)-CHI,
                    Nc_AlGaAs(x)/NCR,
                    Nv_AlGaAs(x)/NVR);

                mat[i][j] = *temp_AlGaAs; // assign the material
                mat[i][j].setDensity(density_of_n);
            }
        }
}
```

```

mat[i][j].setAppliedVolt(vn + i*(vp-vn)/max_i+1);
mat[i][j].set_Diffusivity();

delete temp_AlGaAs; // remove the dummy
}

if ((i>=41) && (i<=122)) // set the rest of the device
if ((i>=61) && (i<=62)) { //set quantum well
    vectx[i]=0;
    mat[i][j] = *temp_GaAs;
    mat[i][j].setDensity(density_of_QW);
    mat[i][j].setAppliedVolt(vn + i*(vp-vn)/max_i+1);
    mat[i][j].set_Diffusivity();
}
else { // set i region
    x =0.3;
    vectx[i]=x;
    Ni_AlGaAs = sqrt(Nc_AlGaAs(x)*Nv_AlGaAs(x))
        * exp(-(Eg_AlGaAs(x)/2)/q)/VT);

    material *temp_AlGaAs = new material(0,0,Ni_AlGaAs,
        eps_AlGaAs(x),
        Eg_AlGaAs(x)-EGAP,
        chi_AlGaAs(x)-CHI,
        Nc_AlGaAs(x)/NCR,
        Nv_AlGaAs(x)/NVR);

    mat[i][j] = *temp_AlGaAs;
    if ((i>=83) && (i<=122)) // set p region
        mat[i][j].setDensity(density_of_p_contact);
    else
        mat[i][j].setDensity(density_of_i);
    mat[i][j].setAppliedVolt(vn + i*(vp-vn)/max_i+1);
    mat[i][j].set_Diffusivity();

    delete temp_AlGaAs;
}
}

// set ohmic contacts at n and p regions
x =0.1; // n-region
Ni_AlGaAs = sqrt(Nc_AlGaAs(x)*Nv_AlGaAs(x))
    * exp(-(Eg_AlGaAs(x)/2)/q)/VT);
material *temp_AlGaAs1 = new material(0,0,Ni_AlGaAs,eps_AlGaAs(x),
    Eg_AlGaAs(x)-EGAP,
    chi_AlGaAs(x)-CHI,
    Nc_AlGaAs(x)/NCR,
    Nv_AlGaAs(x)/NVR);

x =0.3; // p-region
Ni_AlGaAs = sqrt(Nc_AlGaAs(x)*Nv_AlGaAs(x))
    * exp(-(Eg_AlGaAs(x)/2)/q)/VT);
material *temp_AlGaAs = new material(0,0,Ni_AlGaAs,eps_AlGaAs(x),
    Eg_AlGaAs(x)-EGAP,
    chi_AlGaAs(x)-CHI,
    Nc_AlGaAs(x)/NCR,
    Nv_AlGaAs(x)/NVR);

for (j=0; j<max_j+2; j++) { // reset boundaries in y direction
    mat[0][j] = *temp_AlGaAs1;
    mat[0][j].setDensity(density_of_n);
    mat[0][j].setAppliedVolt(vn);
    mat[0][j].set_Diffusivity();

    mat[max_i+1][j] = *temp_AlGaAs;
    mat[max_i+1][j].setDensity(density_of_p_contact);
    mat[max_i+1][j].setAppliedVolt(vp);
    mat[max_i+1][j].set_Diffusivity();
}

delete temp_GaAs;
delete temp_AlGaAs;
delete temp_AlGaAs1;

for (i=1; i<=max_i; i++) { // reset boundaries in x direction
    mat[i][0] = mat[i][2];
    mat[i][max_j+1] = mat[i][max_j-1];
}

```

```

}

// main loop

double old_sum;
double sum = 0;
int round = 1;
double temp_fermi=0;

for (i=1; i<=max_i; i++)
    for(j=1; j<= max_j; j++)
        sum += mat[i][j].potential();

do {    // loop until converge

    old_sum = sum;
    sum = 0;

    for (i=1; i<=max_i; i++) {        // reset boundaries
        mat[i][0] = mat[i][2];
        mat[i][max_j+1] = mat[i][max_j-1];
    }

    for (i=1; i<=max_i; i++) // computing phi
        for(j=1; j<= max_j; j++)
            sum += mat[i][j].cal_phi(mat[i+1][j],mat[i-1][j],
                                     mat[i][j+1],mat[i][j-1]);

    for (i=1; i<=max_i; i++) // computing U
        for(j=1; j<=max_j; j++) {
            mat[i][j].cal_U(mat[i+1][j],mat[i-1][j],mat[i][j+1],
                           mat[i][j-1]);
            mat[max_i-i+1][max_j-j+1].cal_U(mat[max_i-i][max_j-j+1],
                                              mat[max_i-i+2][max_j-j+1],
                                              mat[max_i-i+1][max_j-j],
                                              mat[max_i-i+1][max_j-j+2]);
        }

    for (i=1; i<=max_i; i++) // computing V
        for(j=1; j<=max_j; j++) {
            mat[i][j].cal_V(mat[i+1][j],mat[i-1][j],mat[i][j+1],
                           mat[i][j-1]);
            mat[max_i-i+1][max_j-j+1].cal_V(mat[max_i-i][max_j-j+1],
                                              mat[max_i-i+2][max_j-j+1],
                                              mat[max_i-i+1][max_j-j],
                                              mat[max_i-i+1][max_j-j+2]);
        }

    for (i=1; i<=max_i; i++) // computing field
        for(j=1; j<=max_j; j++) {
            mat[i][j].cal_field(mat[i+1][j],mat[i-1][j],mat[i][j+1],
                               mat[i][j-1]);
            mat[i][j].set_Diffusivity();
        }

    for (i=61; i<=62; i++) // computing quantum wells
        for(j=1; j<=max_j; j++) {
            temp_fermi=mat[i][j].ret_Fermi_leveln(0.3);
            mat[i][j].cal_qwn(temp_fermi,15e-8,0.3);
            temp_fermi=mat[i][j].ret_Fermi_levelp( 0.3);
            mat[i][j].cal_qwp(temp_fermi,15e-8,0.3);
        }

    if ((round%5000) == 0) { // intermediate writing data every 5k iterations

        ddmdata.open("d60nobi_2.dat");
        if (!ddmdata) {
            cout << "Error open 'd60nobi_2.dat'.\n";
            exit(1);
        }

        // write header
        ddmdata << "# this is data of round : " << round << "\n";
        ddmdata << "# |old_sum - sum| = " << fabs(old_sum-sum) << "\n"
                << "\n";

        // write data

```

```

    for (i=1; i<=max_i; i++) {
        for(j=1; j<=max_j; j++)
            ddmdata << i << " "
                << j << " "
                << -mat[i][j].potential()
                << -CHI/q-mat[i][j].CHI()/q << " "
                << -mat[i][j].potential()
                << -CHI/q-mat[i][j].CHI()/q
                << -mat[i][j].EG()/q-EGAP/q << " "
                << mat[i][j].ret_n() << " "
                << mat[i][j].ret_p() << " "
                << mat[i][j].potential() << " "
                << mat[i][j].ret_Fermi_leveln(vectx[i])/q
                << " "
                << mat[i][j].ret_Fermi_levelp(vectx[i])/q
                << " "
                << -mat[i][j].potential()
                << -(mat[i][j].CHI()+CHI)/q
                << -(mat[i][j].EG()+EGAP)/q/2
                << +3/4*VT*log(0.45*9.108e-31/0.067*9.108e-31)
                << "\n";
            ddmdata << "\n";
        }
        ddmdata << "\n\n";
        ddmdata.close();
    }

    round++;

} while(fabs(old_sum-sum) > 1e-5); // within the error?

//**** end cycle ****

ddmdata.open("d60nobi_2.dat"); // writing data

if (!ddmdata) {
    cout << "Error open 'd60nobi_2.dat'.\n";
    exit(1);
}

// header
ddmdata << "# Finished running, round = " << round << "\n";
ddmdata << "# |old_sum - sum| = " << fabs(old_sum-sum) << "\n\n";

// data
for (i=1; i<=max_i; i++) {
    for(j=1; j<=max_j; j++)
        ddmdata << i << " "
            << j << " "
            << -mat[i][j].potential() - CHI/q-mat[i][j].CHI()/q
            << " "
            << -mat[i][j].potential() - CHI/q-mat[i][j].CHI()/q
            << -mat[i][j].EG()/q - EGAP/q << " "
            << mat[i][j].ret_n() << " "
            << mat[i][j].ret_p() << " "
            << mat[i][j].potential() << " "
            << mat[i][j].ret_Fermi_leveln(vectx[i])/q << " "
            << mat[i][j].ret_Fermi_levelp(vectx[i])/q << " "
            << -mat[i][j].potential() - (mat[i][j].CHI()+CHI)/q
            << - (mat[i][j].EG()+EGAP)/q/2
            << + 3/4*VT*log(0.45*9.108e-31/0.067*9.108e-31) << "\n";

        ddmdata << "\n";
    }

    ddmdata << "\n\n";
    ddmdata.close();
}

// function declaration
double eps_AlGaAs( double x) {
    if ((x>=0) && (x<=1))
        return 8.85e-14 * (13.18 - 3.12*x);
    else {
        cout << "x must be between 0.0-1.0\n";
    }
}

```

```

        exit(1);
    }
}

double Eg_AlGaAs( double x) {
    double Eg;
    if ((x >= 0) && (x <= 1)) {
        if ( x <=0.45 )
            Eg = (1.4224 - 1.247*x) * 1.6e-19;
        else
            Eg = (1.900 + 0.125*x + 0.143*x*x)*1.6e-19;
    }
    else {
        cout << "x must be between 0.0-1.0\n";
        exit(1);
    }

    return Eg;
}

double chi_AlGaAs( double x) {
    double chi;
    if ((x>=0) && (x<=0.45))
        chi = (4.07 -0.7482 * x)* 1.6e-19;
    else {
        cout << "x must be between 0.0-0.45\n";
        exit(1);
    }

    return chi;
}

double Nc_AlGaAs(double x) {
    double Nc;
    if ((x>=0) && (x<=0.45))
        Nc = 2.5e19*(0.067+0.083*x);
    else {
        cout << "x must be between 0.0-0.45\n";
        exit(1);
    }

    return Nc;
}

double Nv_AlGaAs(double x) {
    double Nv;
    if ((x>=0) && (x<=0.45))
        Nv = 2.5e19*(0.48+0.31*x);
    else {
        cout << "x must be between 0.0-0.45\n";
        exit(1);
    }

    return Nv;
}

```

เนื่องจากโปรแกรมบน PC เป็นโปรแกรมชนิด Graphical User Interface (GUI) ซึ่งประกอบด้วย ส่วนที่เป็น Resources ต่างๆ จำนวนมาก ดังนั้นจึงขอแสดงเฉพาะส่วนของ Class ที่เกี่ยวข้องกับการ แสดงผล

```
// Material.h: interface for the CMaterial class.
//
///////////////////////////////////////////////////////////////////

#ifdef !defined(AFX_MATERIAL_H_DFEF2D5F_B71A_4201_B479_C270EABE9B2A_INCLUDED_)
#define AFX_MATERIAL_H_DFEF2D5F_B71A_4201_B479_C270EABE9B2A_INCLUDED_

#ifdef _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

class CMaterial
{
protected:
    double phi; // potential
    double field; // electric field
    double U; // special variable for electrons
    double V; // special variable for holes
    double n; // electron concentration
    double p; // hole concentration

    double density; // density of the material
    double phi0; // initial potential
    double phi_applied; // applied potential
    double ni; // ni of the material
    double q; // electron charge
    double eps; // material permittivity
    double Eg; // material bandgap
    double chi; // electron affinity of the material
    double VT; // kT/q
    double Temperature; // Temperature
    double x_right; // hi,j
    double x_left; // hi-1,j
    double y_top; // ki,j
    double y_bottom; // ki,j-1
    double Nc; // effective density of states in conduction band
    double Nv; // effective density of states in valence band
    double Tp; // hole lifetime
    double Tn; // electron lifetime
    double Dn; // electron diffusion coefficient
    double Dp; // hole diffusion coefficient
    double me; // electron mass
    double mh; // hole mass

public:
    // default constructor
    CMaterial();

    // main constructor
    CMaterial(double density_p, double phi_applied_p = 0.0,
        double ni_p = 5915256.5520340, double eps_p = 1.16643e-12,
        double Eg_p = 0, double chi_p = 0,
        double Nc_p = 1, double Nv_p = 1,
        double Temperature_p = 300, double q_p = 1.602e-19,
        double VT_p = 0.026, double x_right_p = 5e-7,
        double x_left_p = 5e-7, double y_top_p = 5e-7,
        double y_bottom_p = 5e-7, double Tp_p = 1e-8,
        double Tn_p = 1e-8, double Dp_p = 10.4, double Dn_p = 221,
        double me_p = 6.103365e-32, double mh_p = 4.099275e-31);
};
```



```

//Set value for variable in inputdata file.
void SetValue(double density_p, double phi_applied_p = 0.0,
    double ni_p = 5915256.5520340, double eps_p = 1.16643e-12,
    double Eg_p = 0, double chi_p = 0,
    double Nc_p = 1, double Nv_p = 1,
    double Temperature_p = 300, double q_p = 1.602e-19,
    double VT_p = 0.026, double x_right_p = 5e-7,
    double x_left_p = 5e-7, double y_top_p = 5e-7,
    double y_bottom_p = 5e-7, double Tp_p = 1e-8,
    double Tn_p = 1e-8, double Dp_p = 10.4, double Dn_p = 221,
    double me_p = 6.103365e-32, double mh_p = 4.099275e-31);

double potential(void) {return phi;} // return
potential
double u(void) {return U;} //
return U
double v(void) {return V;} //
return V
double EG(void) {return Eg;} //
return Eg
double CHI(void) {return chi;} //
return chi
double ret_n(void) {return n;} //
return n
double ret_p(void) {return p;} //
return p
double ret_Ni(void) {return ni;} //
return ni
double ret_Nc(void) {return Nc;} //
return Nc
double ret_Nv(void) {return Nv;} //
return Nv
double ret_VT(void) {return VT;} //
return VT
double ret_Dn(void) {return Dn;} //
return Dn
double ret_Dp(void) {return Dp;} //
return Dp
double ret_me(void) {return me;} //
return me
double ret_mh(void) {return mh;} //
return mh
double ret_x_left(void) {return x_left;} // return
x_left
double ret_x_right(void) {return x_right;} // return
x_right
double ret_y_top(void) {return y_top;} //
return y_top
double ret_y_bottom(void) {return y_bottom;} // return y_bottom
double ret_eps(void) {return eps;} //
return eps
double ret_Field(void) {return field;} //
return field
double ret_Density(void) {return density;} // return
density
double ret_PhiApplied(void) {return phi_applied;} // return phi_applied

// calculate phi from the neighbour's objects
double cal_phi (CMaterial &ipl, CMaterial &iml, CMaterial &jpl,
    CMaterial &jml, double error_limit = 1e-7, int m = 1000);

//calculate U from the neighbour 's objects
double cal_U (CMaterial &ipl, CMaterial &iml, CMaterial &jpl, CMaterial &jml);

//calculate V from the neighbour 's objects
double cal_V (CMaterial &ipl, CMaterial &iml, CMaterial &jpl, CMaterial &jml);

// calculate electron quasi-Fermi level
double cal_fermi_level_n(double ref_chi, double ref_Eg);

// calculate hole quasi-Fermi level
double cal_fermi_level_p(double ref_chi, double ref_Eg);

// calculate electron density in QW
void cal_qwn(double width, double depth, double ref_Eg, double ref_chi);
// width is in meter and depth is in Joule

```

```

// calculate hole density in QW
void cal_qwp(double width, double depth, double ref_Eg, double ref_chi);
// width is in meter and depth is in Joule

//calculate field from the neighbour's objects
void cal_field (CMaterial &ipl, CMaterial &iml, CMaterial &jpl, CMaterial
&jml);

void set_Density(double denst); //
set material density
void set_AppliedVolt(double phi_appl); // set the
applied voltage
void set_time(double tau); //
set lifetime
void set_x_left(double sizel) {x_left = sizel;} // set x_left
void set_x_right(double sizel) {x_right = sizel;} // set x_right
void set_y_top(double sizel) {y_top = sizel;} // set y_bottom
void set_y_bottom(double sizel) {y_bottom = sizel;} // set y_bottom
CMaterial operator= (const CMaterial &input); // declare equal sige
output

virtual ~CMaterial();

};

#endif // !defined(AFX_MATERIAL_H__DFEF2D5F_B71A_4201_B479_C270EABE9B2A__INCLUDED_)

```

● Class Implementation (Material.cpp)

```

//
/////////////////////////////////////////////////////////////////

#include "stdafx.h"

#include <iostream.h>
#include <math.h>
#include <stdlib.h>

#include "Self_Consistence.h"
#include "Material.h"

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#define new DEBUG_NEW
#endif

/////////////////////////////////////////////////////////////////
// Construction/Destruction
/////////////////////////////////////////////////////////////////

CMaterial::CMaterial()
{
    phi = 0;
    field = 0;
    phi0 = 0;
    phi_appl = 0;
    U = 0;
    V = 0;
    n = 0;
    p = 0;
    density = 0;
    ni = 0;
    q = 0;
    eps = 0;
    Eg = 0;
    chi = 0;
    VT = 0;
    Temperature = 0;
    x_right = 0;
    x_left = 0;
    y_top = 0;
    y_bottom = 0;
    Nc = 0;
}

```

```

Nv          = 0;
Tp          = 0;
Tn          = 0;
Dp          = 0;
Dn          = 0;
}

CMaterial::CMaterial(double density_p, double phi_applied_p,
                    double ni_p, double eps_p, double Eg_p,
                    double chi_p, double Nc_p, double Nv_p,
                    double Temperature_p, double q_p,
                    double VT_p, double x_right_p,
                    double x_left_p, double y_top_p,
                    double y_bottom_p, double Tp_p,
                    double Tn_p, double Dp_p, double Dn_p,
                    double me_p, double mh_p)
{
    double temp;
    density          = density_p;
    phi_applied      = phi_applied_p;
    ni               = ni_p;
    q                = q_p;
    eps              = eps_p;
    Eg               = Eg_p;
    chi              = chi_p;
    VT               = VT_p;
    Temperature      = Temperature_p;
    x_right           = x_right_p;
    x_left           = x_left_p;
    y_top            = y_top_p;
    y_bottom         = y_bottom_p;
    Nc                = Nc_p;
    Nv                = Nv_p;
    Tp                = Tp_p;
    Tn                = Tn_p;
    Dp                = Dp_p;
    Dn                = Dn_p;
    field            = 0;
    me                = me_p;
    mh                = mh_p;

    if (density<0) {
        temp = -density / (2 * ni );
        phi0 = -log(temp + sqrt((temp*temp) + 1));
    }
    else {
        temp = density / ( 2 * ni);
        phi0 = log(temp + sqrt((temp * temp ) + 1 ));
    }

    phi = phi_applied + phi0*VT;
    U   = exp(-phi_applied/VT);
    V   = exp(phi_applied/VT);
    //n  = ni * Nc * U * exp(phi/VT) * exp(chi/q/VT);
    n   = ni * Nc * U * exp((phi+chi/q)/VT);
    //p  = ni * Nv * V * exp(-phi/VT) * exp(-(chi+Eg)/q/VT);
    p   = ni * Nv * V * exp(-(phi+(chi+Eg)/q)/VT);
}

void CMaterial::SetValue(double density_p, double phi_applied_p,
                        double ni_p, double eps_p,
                        double Eg_p, double chi_p,
                        double Nc_p, double Nv_p,
                        double Temperature_p, double q_p,
                        double VT_p, double x_right_p,
                        double x_left_p, double y_top_p,
                        double y_bottom_p, double Tp_p,
                        double Tn_p, double Dp_p, double Dn_p,
                        double me_p, double mh_p)
{
    double temp;
    density          = density_p;
    phi_applied      = phi_applied_p;
    ni               = ni_p;
    q                = q_p;
    eps              = eps_p;
    Eg               = Eg_p;

```

```

chi          = chi_p;
VT           = VT_p;
Temperature  = Temperature_p;
x_right      = x_right_p;
x_left       = x_left_p;
y_top        = y_top_p;
y_bottom     = y_bottom_p;
Nc           = Nc_p;
Nv           = Nv_p;
Tp           = Tp_p;
Tn           = Tn_p;
Dp           = Dp_p;
Dn           = Dn_p;
field        = 0;
me           = me_p;
mh           = mh_p;

if (density<0) {
    temp = -density / (2 * ni);
    phi0 = -log(temp + sqrt((temp*temp) + 1));
}
else {
    temp = density / (2 * ni);
    phi0 = log(temp + sqrt((temp * temp) + 1));
}

phi = phi_applied + phi0*VT;
U   = exp(-phi_applied/VT);
V   = exp(phi_applied/VT);
n   = ni * Nc * U * exp((phi+chi/q)/VT);
p   = ni * Nv * V * exp(-(phi+(chi+Eg)/q)/VT);
}

double CMaterial::cal_phi(CMaterial &ipl, CMaterial &iml, CMaterial &jpl,
                          CMaterial &jml, double error_limit, int
m)
{
    double y, dy, dxeps, dyeps;
    double aij, aiplj, aimlj, aijpl, aijml;
    double F, f;
    double G1, G2;
    double dfx, dfy;

    if (m <= 0)
    {
        cout << "It's not funny" << m << ", set 'm' again.\n";
        exit(1);
    }

    dxeps = ipl.eps/x_right/2 - iml.eps/x_left/2 - eps*(1/x_right-1/x_left)/2;
    dyeps = jpl.eps/y_top/2 - jml.eps/y_bottom/2 - eps*(1/y_top-1/y_bottom)/2;
    aiplj = eps^2./((x_right+x_left)/x_right + dxeps/x_right/2;
    aimlj = eps^2./((x_right+x_left)/x_left - dxeps/x_left/2;
    aijpl = eps^2./((y_top+y_bottom)/y_top + dyeps/y_top/2;
    aijml = eps^2./((y_top+y_bottom)/y_bottom-dyeps/y_bottom/2;
    aij    = aiplj - aimlj + aijpl + aijml;

    f = -q*density;
    F = aiplj*ipl.phi + aimlj*iml.phi + aijpl*jpl.phi + aijml*jml.phi - f;
    G1 = q*ni*(Nc*U*exp((phi+chi/q)/VT)
            ~ Nv*V*exp(-(phi+(chi+Eg)/q)/VT));
    y = aij*phi + G1 - F;

    while ((fabs(y) > error_limit) && (m > 0))
    {
        m--;
        G2 = q*ni*(Nc*U/VT*exp((phi+chi/q)/VT)
            + Nv*V/VT*exp(-(phi+(chi+Eg)/q)/VT));
        dy = aij + G2;
        phi -= y/dy;

        G1 = q*ni*(Nc*U*exp((phi+chi/q)/VT)
            ~ Nv*V*exp(-(phi+(chi+Eg)/q)/VT));
        y = aij*phi + G1 - F;
    }
}

```

```

if ( m < G )
{
    cout << "dy = " << dy << "\n";
    cout << "1/dy = " << 1/dy << "\n";
    cout << "Function is not convergent , exit program. \n";
    exit(1);
}

n = ni * Nc * U * exp((phi+chi/q)/VT);
p = ni * Nv * V * exp(-(phi+(chi+Eg)/q)/VT);

dfx = -(ipl.phi-phi)/x_right/2 - (phi-impl.phi)/x_left/2;
field = sqrt(dfx*dfx);

dfx = -(ipl.phi-phi)/x_right/2 - (phi-impl.phi)/x_left/2;
dfy = -(jpl.phi-phi)/y_top/2 - (phi-jml.phi)/y_bottom/2;
field = sqrt(dfx*dfx + dfy*dfy);

return phi;
}

double CMaterial::cal_U(CMaterial &ipl, CMaterial &iml, CMaterial &jpl,
                        CMaterial &jml)
{
    double aij, aiplj, aimlj, aijpl, aijml;
    double cnhp, cnhm, cnkp, cnkm, bnhp, bnhm, bnkp, bnkm;
    double enhp, xnhp, xnhm, xnkp, xnkml;
    double F, f, G, r;

    cnhp = 2*Dn*Nc / (x_right+x_left) / x_right;
    xnhp = (phi-ipl.phi)/VT + (chi-ipl.chi)/q/VT;
    bnhp = fabs(xnhp) < 1e-10 ? 1-0.5*xnhp : xnhp/(exp(xnhp)-1);
    aiplj = cnhp * bnhp;

    cnhm = 2*Dn*Nc / (x_right+x_left) / x_left;
    xnhm = (phi-impl.phi)/VT + (chi-impl.chi)/q/VT;
    bnhm = fabs(xnhm) < 1e-10 ? 1-0.5*xnhm : xnhm/(exp(xnhm)-1);
    aimlj = cnhm * bnhm;

    cnkp = 2*Dn*Nc / (y_top+y_bottom) / y_top;
    xnkp = (phi-jpl.phi)/VT + (chi-jpl.chi)/q/VT;
    bnkp = fabs(xnkp) < 1e-10 ? 1-0.5*xnkp : xnkp/(exp(xnkp)-1);
    aijpl = cnkp * bnkp;

    cnkm = 2*Dn*Nc / (y_top+y_bottom) / y_bottom;
    xnkml = (phi-jml.phi)/VT + (chi-jml.chi)/q/VT;
    bnkml = fabs(xnkml) < 1e-10 ? 1-0.5*xnkml : xnkml/(exp(xnkml)-1);
    aijml = cnkm * bnkml;

    aij = aiplj + aimlj + aijpl + aijml;

    enhp = exp((phi+chi/q)/VT);

    r = 1. / (Tn*ni*(Nv*V*exp(-(phi+chi/q)/VT)+1)
              + Tp*ni*(Nc*U*exp((phi+chi/q)/VT)+1));

    G = ni*V*r/enhp;
    f = -ni*r/enhp;
    F = aiplj*ipl.U + aimlj*iml.U + aijpl*jpl.U + aijml*jml.U - f;

    U = F / (aij*G);

    n = ni*Nc*U*exp((phi+chi/q)/VT);

    return U;
}

double CMaterial::cal_V(CMaterial &ipl, CMaterial &iml, CMaterial &jpl,
                        CMaterial &jml)
{
    double aij, aiplj, aimlj, aijpl, aijml;
    double cphp, cphm, cpkp, cpkm, bphp, bphm, bpkp, bpkml;
    double ephp, xphp, xphm, xpkp, xpkml;
    double F, f, G, r;

    cphp = 2*Dp*Nv / (x_right+x_left) / x_right;

```

```

xphp = (ipl.phi-phi)/VT + (ipl.chi-chi)/q/VT + (ipl.Eg-Eg)/q/VT;
bphp = fabs(xphp) < 1e-10 ? 1+0.5*xphp : xphp/(exp(xphp)-1);
aiphj = cphp * bphp ;

cphm = 2*Dp*Nv / (x_right+x_left) / x_left;
xphm = (iml.phi-phi)/VT + (iml.chi-chi)/q/VT + (iml.Eg-Eg)/q/VT;
bphm = fabs(xphm) < 1e-10 ? 1+0.5*xphm : xphm/(exp(xphm)-1);
aimlj = cphm * bphm ;

cpkp = 2*Dp*Nv / (y_top+y_bottom) / y_top;
xpkp = (jpl.phi-phi)/VT + (jpl.chi-chi)/q/VT + (jpl.Eg-Eg)/q/VT;
bpkp = fabs(xpkp) < 1e-10 ? 1+0.5*xpkp : xpkp/(exp(xpkp)-1);
aijpl = cpkp * bpkp;

cpkm = 2*Dp*Nv / (y_top+y_bottom) / y_bottom;
xpkm = (jml.phi-phi)/VT + (jml.chi-chi)/q/VT + (jml.Eg-Eg)/q/VT;
bpkm = fabs(xpkm) < 1e-10 ? 1+0.5*xpkm : xpkm/(exp(xpkm)-1);
aijml = cpkm * bpkm;

aij = aiphj + aimlj + aijpl + aijml;

ephp = exp(-(phi+chi/q+Eg/q)/VT);

r = 1. / (Tn*ni*(Nv*V*exp(-(phi+(chi+Eg)/q)/VT)+1)
          + Tp*ni*(Nc*U*exp((phi+chi/q)/VT)+1));

G = ni*U*r/ephp;
f = -ni*r/ephp;
F = aiphj*ipl.V + aimlj*iml.V + aijpl*jpl.V + aijml*jml.V - f;
V = F / (aij+G);

p = ni*Nv*V*exp(-phi/VT)*exp(-(chi+Eg)/q/VT);

return V;
}

void CMaterial::cal_field(CMaterial &ipl, CMaterial &iml, CMaterial &jpl,
                        CMaterial &jml)
{
    double dfx, dfy;

    dfx = -(ipl.phi-phi)/x_right/2 - (phi-iml.phi)/x_left/2;
    dfy = -(jpl.phi-phi)/y_top/2 - (phi-jml.phi)/y_bottom/2;

    field = sqrt(dfx*dfx + dfy*dfy);
}

double CMaterial::cal_fermi_level_n(double ref_chi, double ref_Eg)
{
    double Efi, Ef;

    Efi = -phi*q - (chi+ref_chi) - (Eg+ref_Eg)/2 + 3/4*VT*q*log(mh/me);

    Ef = Efi - q*(phi + chi/q + VT*log(Nc) + VT*log(U));
    return Ef;
}

double CMaterial::cal_fermi_level_p(double ref_chi, double ref_Eg)
{
    double Efi, Ef;

    Efi = -phi*q - (chi+ref_chi) - (Eg+ref_Eg)/2 + 3/4*VT*q*log(mh/me);

    Ef = Efi - q*(phi + chi/q + Eg/q - VT*log(Nv) - VT*log(V));

    return Ef;
}

void CMaterial::cal_qwn(double width, double depth, double ref_Eg, double ref_chi)
// width is in meter and depth is in Joule
{
    int num,i;
    double tmp = 0;
    double gam, hbar, lf, rg, a_temp, y, Ef;
    double Energ[30];
    double a[30];

```

```

Ef = cal_fermi_level_n(ref_chi, ref_Eg);

hbar = 6.625e-34/(2*3.14159);
gam = sqrt(2*(me/hbar)*depth*width*width/hbar);

num=0;

do
{
    lf=0;
    rg=1;
    num++;

    do
    {
        a_temp = (lf+rg)/2;
        y = gam*a_temp-((num-1)*3.14159+2*acos(a_temp));

        if (y<0)
            lf=a_temp;
        else
            rg=a_temp;
    } while((fabs(y)>1e-5)&&((1-lf)>1e-8));

    Energ[num]-depth*a_temp*a_temp;
    a[num]=a_temp;

} while(((1-lf)>1e-8) && (num<2));

n=0;

for(i=1;i<num;i++)
{
    tmp = log(1-exp((Ef+phi*q+(chi+ref_chi)-Energ[i])/(VT*q)));
    if (tmp==0)
        tmp = exp((Ef+phi*q+(chi+ref_chi)-Energ[i])/(VT*q));
    n += 7.232e+15*tmp;
}

U = exp(-(phi+chi/q)/VT)* n/(ni*Nc);
}

void CMaterial::cal_qwp(double width, double depth, double ref_Eg, double ref_chi)
// width is in meter and depth is in Joule
{
    int num, i;
    double tmp;

    double gam, hbar, lf, rg, a_temp, y, Ef;
    double Energ[30];
    double a[30];

    Ef = cal_fermi_level_p(ref_chi, ref_Eg);

    hbar = 6.625e-34/(2*3.14159);
    gam = sqrt(2*me*depth*width*width/(hbar*hbar));
    num = 0;

    do
    {
        lf = 0;
        rg = 1;
        num++;

        do
        {
            a_temp = (lf+rg)/2;
            y = gam*a_temp - ((num-1)*3.14159+2*acos(a_temp));

            if (y<0)
                lf=a_temp;
            else
                rg=a_temp;
        } while((fabs(y)>1e-5)&&((1-lf)>1e-8));
    }

```

```

    Energ[num] = depth*a_temp*a_temp;
    a[num] = a_temp;

    while(((1-1f)>1e-8)&&(num<2));

    p = 1.0e-10;

    for(i=1;i<num;i++) {
        tmp=log(1+exp((-phi*q-(chi+ref_chi)-(Eg+ref_Eg)-Energ[i]-Ef)/(VT*q)));
        if (tmp==0)
            tmp = exp((-phi*q-(chi+ref_chi)-(Eg+ref_Eg)-Energ[i]-Ef)/(VT*q));
        p += 4.857e+16*tmp;
    }

    V = p/(ni*Nv) * exp((phi+(chi+Eg)/q)/VT);
}

void CMaterial::set_Density(double denst)
{
    double temp;

    density = denst;

    if (density<0)
    {
        temp = -density/2*ni;
        phi0 = -log(temp + sqrt((temp*temp) + 1));
    }
    else
    {
        temp = density/2*ni;
        phi0 = log(temp + sqrt((temp*temp) + 1));
    }

    phi = phi_applied + phi0*VT;
    n = ni*Nc*U*exp((phi+chi/q)/VT);
    p = ni*Nv*V*exp(-(phi+(chi+Eg)/q)/VT);
}

void CMaterial::set_AppliedVolt(double phi_appl)
{
    phi_applied = phi_appl;
    phi = phi_applied + phi0*VT;
    U = exp(-phi_applied/VT);
    V = exp(phi_applied/VT);

    n = ni*Nc*U*exp((phi+chi/q)/VT);
    p = ni*Nv*V*exp(-(phi+(chi+Eg)/q)/VT);
}

void CMaterial::set_time(double tau)
{
    Tp = tau;
    Tn = tau;
}

CMaterial CMaterial::operator=(const CMaterial &input)
{
    phi = input.phi;
    field = input.field;
    U = input.U;
    V = input.V;
    phi0 = input.phi0;
    phi_applied = input.phi_applied;
    density = input.density;
    ni = input.ni;
    n = input.n;
    p = input.p;
    q = input.q;
    eps = input.eps;
    Eg = input.Eg;
    chi = input.chi;
    VT = input.VT;
    Temperature = input.Temperature;
    x_right = input.x_right;
    x_left = input.x_left;
}

```



```

    y_top          = input.y_top;
    y_bottom       = input.y_bottom;
    Tp             = input.Tp;
    Tn             = input.Tn;
    Dn             = input.Dn;
    Dp             = input.Dp;
    Nc             = input.Nc;
    Nv             = input.Nv;
    me             = input.me;
    mh             = input.mh;

    return *this;
}

CMaterial::~CMaterial()
{
}

```

ภาคผนวก จ.
ผลงานในการประชุมวิชาการ

ผลงานในการประชุมวิชาการ ณ งาน วทท. 28 ในเดือนตุลาคม 2545

การประยุกต์ใช้ปรากฏการณ์ควอนตัมคอนไฟน์เมนต์ใน แหล่งกำเนิดแสงหลายความยาวคลื่นชนิดใหม่

AN APPLICATION OF THE QUANTUM CONFINED STARK'S EFFECT IN A NOVEL MULTIPLE WAVELENGTH LIGHT EMITTER.

ภัทร อัยรักษ์¹, บุญเหลือ พงศ์ดารา¹, แอนโทนี วิกเกอร์ส²

Pattara Aiyarak¹, Boonlua Phongdara¹, Anthony Vickers²

¹Department of Physics, Faculty of Science, Prince of Songkla University, Hatyai, Songkla 90112, Thailand; ²Department of Electrical System Engineering, University of Essex, Colchester, CO4 3SQ, United Kingdom; e-mail address:

apattara@ratree.psu.ac.th

บทคัดย่อ: แหล่งกำเนิดแสงหลายความยาวคลื่นชนิดสารกึ่งตัวนำซึ่งประกอบด้วยหลุมควอนตัม ได้ถูกออกแบบขึ้นเพื่อศึกษาความเป็นไปได้โดยใช้คอมพิวเตอร์เป็นเครื่องมือในการศึกษา โครงสร้างของสิ่งประดิษฐ์สารกึ่งตัวนำนี้ใช้ $\text{Al}_x\text{Ga}_{1-x}\text{As-GaAs}$ เป็นโครงสร้างดังแสดงในรูปที่ 1 โดยมีกำแพงของระดับพลังงานที่รอยต่อของ n-doped และ intrinsic ซึ่งใน intrinsic นี้มีหลุมควอนตัมอยู่ตรงกลาง เมื่อถูก forward bias โสไลต์ p-doped จะเคลื่อนที่มารออยู่ที่หลุมควอนตัม อย่างไรก็ตามอิเล็กตรอนไม่สามารถเข้ามาพบกับโฮลได้เนื่องจากการเคลื่อนที่ของอิเล็กตรอนถูกกั้นโดยกำแพงพลังงานที่รอยต่อของ n-doped และ intrinsic อิเล็กตรอนจะเคลื่อนที่เข้ามาได้ก็ต่อเมื่อบริเวณ n-doped มีสนามไฟฟ้าแบบขวางมาให้พลังงาน หรือ "heat" อิเล็กตรอนทำให้อิเล็กตรอนมีพลังงานมากพอที่จะข้ามกำแพงพลังงานแล้วเข้าไปที่หลุมควอนตัมเพื่อรวมตัวกับโฮล แล้วให้แสงออกมา ขนาดของการ forward bias เป็นตัวควบคุมความยาวคลื่นที่จะออกมาเนื่องจากขนาดของการ forward bias ที่ต่างกันจะทำให้มีสนามไฟฟ้าต่างกันเกิดขึ้นที่บริเวณหลุมควอนตัม และเนื่องจากปรากฏการณ์ควอนตัมคอนไฟน์เมนต์ ระดับพลังงานในหลุมควอนตัมจึงต่างกันเป็นผลให้ได้แสงที่มีความยาวคลื่นที่ต่างกัน คอมพิวเตอร์ได้ถูกนำมาศึกษาความสัมพันธ์ระหว่าง forward bias ที่ขนาดต่างๆ กับความยาวคลื่นที่ของแสงที่ออกมา

Abstract: A quantum well semiconductor light emitter was designed and simulated by a computer in order to realize a possibility of being a multiple wavelength light emitter. The structure, as shown in Figure 1, is an $\text{Al}_x\text{Ga}_{1-x}\text{As-GaAs}$ system with an energy barrier at the interface between the n-doped region and the intrinsic region. The GaAs quantum well, in addition, is situated at the middle of the intrinsic region. When the device is forward biased, the holes from the p-doped region drift into the quantum well and wait for the electrons from the n-doped region; however, the electrons from the n-doped region is prevented from entering the quantum well because of the energy barrier. Once the n-doped region is applied with a transverse electric field, the electrons are so-called heated and able to overcome the energy barrier. The electrons, therefore, enter the quantum well and recombine with the holes; hence the light. The wavelength of the light can be altered by changing the forward bias voltage. The difference in the forward bias voltage gives the different electric field across the quantum well. Because of the quantum confined Stark's effect, the energy levels in the quantum well are different which correspond to the different emitting wavelength. In order to estimate the emitting wavelength with a certain forward bias, a computer simulation is required.

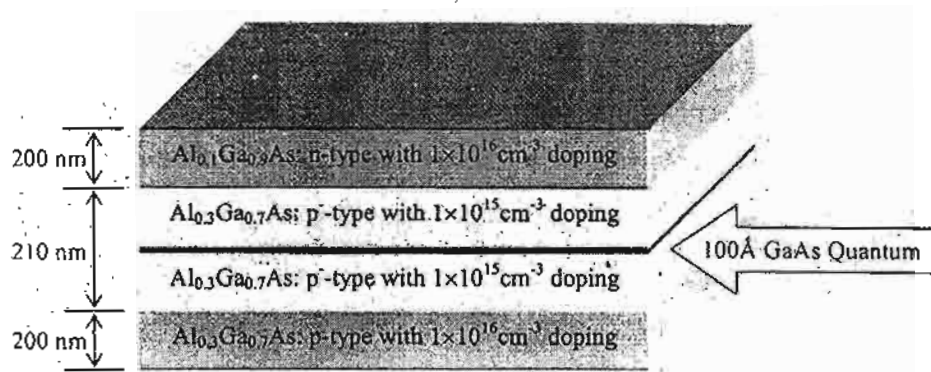


Figure 1: The structure of the device.

Methodology: A computer program was written in order to simulate the device. Poisson and Schrödinger's equations were solved self-consistently in the quantum well while the bulk material used the drift-diffusion model to simulate. Ultimately, the program will be used to study the relationship between the forward bias and the emitting wavelength after the n-doped region is heated. The program was written with the object-oriented concept so that the program is easy to modify and upgrade in the future.

Results, Discussion and Conclusion: The program is divided into two parts, i.e. drift-diffusion and quantum model. The first part is finished and verified by simulating a simple device, e.g. p-n junction diode, and the result fits with the analytical result. However, once the models were combined, the convergent problem occurred. The program and parameters are being studied and modified in order to fix this problem.

References: (i) Vickers, A. J., Mashayekhi, H. R., Aiyarak, P. and Oriato, D. (1999) *Turkish Journal of Physics*. **23**, 649.

Keywords: quantum confined Stark's effect, multiple wavelength light emitter, simulation, self-consistence

ภาคผนวก จ.
Manuscript

A Computational Search for the Optimized Structure of a Novel Light Emitter

P. Aiyarak¹, A. J. Vickers² and B. Phongdara¹

¹Department of Physics, Faculty of Science, Prince of Songkla University, Hatyai, Songkhla, 90112, Thailand

²Department of ESE, University of Essex, Colchester, CO4 3SQ, UK.

Abstract

A computer simulation is required when a novel idea of the semiconductor device was proposed. The device, which is called Hot Electron Barrier Light Emitter or HEBLE, being simulated here is a light source with the potential for being a multi-wavelength light emitter in the WDM system. The structure of the device is based on the GaAs/Al_xGa_{1-x}As material system because it is well understood. The novelty of this device is in the way it operates. The difference in this device is the energy barrier at the interface between the n-doped region and the intrinsic region. The GaAs quantum well, in addition, is situated at the middle of the intrinsic region. When the device is forward biased, the holes from the p-doped region drift into the quantum well and wait for the electrons from the n-doped region; however, the electrons from the n-doped region is prevented from entering the quantum well because of the energy barrier. Once the n-doped region is applied with a transverse electric field, the electrons are so-called heated and able to overcome the energy barrier. The electrons, therefore, enter the quantum well and recombine with the holes; hence the light. The wavelength of the light can be altered by changing the forward bias voltage. The difference in the forward bias voltage gives the different electric field across the quantum well. Because of the quantum confined Stark's effect, the energy levels in the quantum well are different which correspond to the different emitting wavelength. To find an appropriate structure for the device, varieties of structures were put into the simulation. The best solution for the device is discussed.

1. Introduction

As multiple wavelength emitters are required in the WDM system, research in this area is very intensive. The tunable lasers can be roughly divided into three groups[1]. The first group is the mechanically tuned lasers. This type of laser is mainly a Fabry-Perot cavity that is adjacent to the lasing medium (i.e. external cavity) to filter out the unwanted wavelengths. Tuning is achieved by adjusting the distance between two mirrors. This type of tunable laser provides a wide range of tunability but the tuning time is limited. The second type of tunable laser is an acousto-optically and electro-optically tuned laser. These are tuned by applying sound waves or electrical current into the external cavity in order to change the refractive index. The tuning time is limited by the time required for light to build up in the cavity at the new frequency. The tuning time of an acousto-optic laser is a lot quicker than the mechanically tuned laser but still slower than the electro-optic laser. The tuning range is limited by the range of frequencies generated by the laser. The vertical cavity surface emitting laser (VCSEL) coupled with a micromachined deformable-membrane has been used to make an electro-optic laser. The wavelength tuning range of the laser is rather narrow, 19.1nm, but it can be tuned continuously[2]. A more recent example of an electro-optic laser is a micro-electromechanical tunable VCSEL laser (MEM-VCSEL)[3].

The device has three terminals where the first two are for biasing the laser and the third one is for changing the wavelength by moving the top mirror; hence the laser cavity. The last group of tunable lasers is totally different from the previous two. Instead of changing the external parameter, this type of tunable laser is tuned by changing the internal parameters of the laser. The parameters can be changed by injecting current into parts of laser or simply changing the bias. This group of tunable lasers has a wider range of wavelength than the second group and the tuning time is quicker than the first group. This type of laser will be discussed, as it is very popular and intensively studied. Several examples of this type of lasers are being studied worldwide. The examples of the laser in this group are the three-terminal bias induced color-tunable emitter (BICE)[4], the three-branch laser or Y-laser[5], the sampled grating distributed Bragg reflector (SG-DBR)[6], the super structure grating distributed Bragg reflector (SSG-DBR)[7], and etc. Here we propose another device in this category. The novelty of the device is the way of changing the wavelength. The device uses the quantum confined Stark's effect (QCSE) to change the wavelength. The structure, as shown in Figure 1, is an $\text{Al}_x\text{Ga}_{1-x}\text{As}$ -GaAs system with an energy barrier at the interface between the n-doped region and the intrinsic region. The GaAs quantum well, in addition, is situated at the middle of the intrinsic region. When the device is forward biased, the holes from the p-doped region drift into the quantum well and wait for the electrons from the n-doped region; however, the electrons from the n-doped region is prevented from entering the quantum well because of the energy barrier. Once the n-doped region is applied with a transverse electric field, the electrons are so-called heated and able to overcome the energy barrier. The electrons, therefore, enter the quantum well and recombine with the holes; hence the light. The wavelength of the light can be altered by changing the forward bias voltage. The difference in the forward bias voltage gives the different electric field across the quantum well. Because of the QCSE, the energy levels in the quantum well are different which correspond to the different emitting wavelength. A self-consistence between the drift-diffusion and Schrödinger equations are required to study the device.

2. Computer model

Our aim was to discover a device that does not turn on when forward biased, unless a transverse electric field is also applied. The simulation was undertaken on devices based on the $\text{Al}_x\text{Ga}_{1-x}\text{As}$ -GaAs p-i-n structure with a quantum well within the i-region. The simulation uses a drift-diffusion transport model which is widely used in solving semiconductor device problems. The model consists of the well-known classical Poisson's equation and current continuity equations for both holes and electrons. The structure of interest contains a quantum well which cannot be solved by classical equations, only a solution of Schrödinger equation gives a correct estimation of electrons and holes in the quantum well. Therefore, a numerical solution of the Schrödinger equation is also required. In order to solve the derivative equations numerically, a particular definition of derivative is used. All main equations in the simulation were solved using the finite difference method and in some cases Newton's method were used in order to solved the non-linear algebraic equations[8]. The simulation is implemented in the C++ language and run under a UNIX environment. Using the technique of object oriented programming provided by C++; the program was written and modified easily[9].

The results from the simulation are the band edge diagram and carrier concentrations across the device at different forward bias voltages. Figure 2 shows the template of

the structure which will be simulated. To find the appropriate structure, parameters were altered and the solutions were compared and discussed.

3. Numerical results and discussion

After the parameters for the first device have been set, the device has been simulated without any voltage applied, i.e. in equilibrium. The band edge diagram can be seen in Figure 3. The electron and hole concentrations at different locations of the device are plotted against the applied voltages in Figure 4. Increasing the applied voltage increases the carrier concentrations in the quantum well and also increases the hole population in the n-doped region. The hole concentration in the quantum well is approximately 10^7 times higher at one-volt forward bias compared to the equilibrium. When the forward bias is increased to two volts, the hole concentration in the quantum well increases as expected. However, it is also high in the n-doped region, which will lead to unwanted recombination and light from the n-doped region. It can also be seen that the barrier at the interface of the n- and p-doped regions does prevent electrons from the n-doped region flooding into the quantum well as the electron concentration in the quantum well is very low with every applied voltage. Many parameters, such as the position of the quantum well, the doping and the dimension of each region, were altered to see whether an operational device could be achieved.

A desired device is a device that does not turn on, i.e. no light emission, when forward biased, unless a transverse electric field is also applied to the n-doped region. In other words, when the device is only forward biased, the electron concentration in the quantum well must be low while the hole concentration in it must be high. However, to avoid bulk emission, i.e. the emission at the n-doped region, the hole concentration in the n-doped region must be low.

The first parameter to be altered is the percentage of aluminium in the p⁻- and p-doped regions. Actually the first device with $\text{Al}_{0.3}\text{Ga}_{0.7}\text{As}$ gives a reasonable result.

However, it is a good idea to look at the effect of the composition of aluminum and the result is shown in Figure 5. A lower composition gives a lower barrier, so that the electrons can be heated easier. The main effect of this parameter is on the height of the heterojunction at the p-n interface and also the confinement of the quantum well. As the composition of Al in the p⁻- and p-doped regions varies from 0.1 to 0.3, we can see that the hole concentration in the quantum well increases, i.e. a better confinement in the quantum well, while it decreases in the n-doped region. At the same time the increasing aluminium concentration produced a dramatic decrease of the electron population in the quantum well. So with reasons, the program gives appropriate results and an idea of how much the aluminium composition in p⁻- and p-doped region should be. Clearly in terms of our requirements we should choose a high aluminium concentration. Therefore $\text{Al}_{0.3}\text{Ga}_{0.7}\text{As}$ was chosen for the p⁻- and p-type regions.

The next parameter to be looked at is the position of the quantum well. The test devices will have the quantum well from 100nm away from the p-doped region to 300nm. Figure 6 shows the carrier concentrations in the quantum well and n-doped region. It can be seen that the closer the quantum well is to the p-doped region the higher the hole concentration in the quantum well at the same bias. The electron concentration in the quantum well is also lower when the quantum well is closer to the p-doped region. The hole concentration in n-doped region is, however, almost constant throughout the variation of the position of the quantum well. It is clear that the optimized device would have the quantum well at 100nm from the p-doped region.

The next parameter to be looked at is the size of the p⁻-doped region. As said earlier, the ideal device would have holes populating the quantum well when it is forward biased alone. If the distance from the n-doped region to the quantum well is long, it might take a longer time for electrons to arrive when the device is electrically heated limiting the response of the device, also losses due to the recombination. The length of the p⁻-doped region was varied in order to find the optimum. The quantum well was fixed at 100nm from the p-doped region.

It has been found, as seen in Figure 7, from the simulation that the results are almost identical in all sizes. As mentioned earlier, the smaller size of p⁻-doped region makes the device more responsive, and, as the simulation shows nothing different between the different sizes, the size of 200nm is chosen.

The next concern is the hole concentration in the quantum well. It seems to be low as the doping of the p-doped region is low. We are trying to have the highest possible concentration of holes in the quantum well by increasing the doping concentration in the p-doped region. However, the drawback is that the hole concentration in n-doped region is going to be higher as well. The variation of the doping concentration in the p-doped region will be from 1×10^{16} to $1 \times 10^{18} \text{ cm}^{-3}$. Figure 8 shows the concentration of holes and electrons in the quantum well and n-doped region. There is a big improvement of the hole concentration in the quantum well from the doping of 1×10^{16} to $1 \times 10^{17} \text{ cm}^{-3}$; however, it seems not to change much between the doping of 1×10^{17} and $1 \times 10^{18} \text{ cm}^{-3}$. The doping of $1 \times 10^{17} \text{ cm}^{-3}$ gave an acceptable result as the hole concentration is approximately 10^{17} cm^{-3} in the quantum well, which is high, but in the n-doped region is approximately 10^{15} cm^{-3} , which is still low. The doping of $1 \times 10^{17} \text{ cm}^{-3}$, therefore, was chosen for the doping of the p-doped region.

The next parameter to be looked at in this simulation is the percentage of the aluminium in the n-doped region. Previously, the n-doped region was GaAs. Increasing the aluminium composition would result in lowering the barrier. The result is shown in Figure 9. The hole concentration in the quantum well does not change with different materials for the n-doped region; however, the hole concentration in the n-doped region changes dramatically. It can also be seen that the electron concentration is getting higher in the quantum well as the aluminium composition of the n-doped region's material is higher, i.e. the barrier is lower. We have chosen $\text{Al}_{0.1}\text{Ga}_{0.9}\text{As}$ to be the material for the n-doped region as it gives a lower hole concentration in n-doped region. The reason that $\text{Al}_{0.2}\text{Ga}_{0.8}\text{As}$ was not chosen is that the heating might be difficult. The mobility of the material will be lower when the aluminium composition is higher, which makes it difficult to heat electrically. Moreover, the electron concentration in the quantum well is getting higher as the barrier is lower.

Last but not least, the doping concentration of the n-doped region was investigated. The reason we used $1 \times 10^{16} \text{ cm}^{-3}$ for the doping of the n-doped region is that we might have encountered problems when heating a high doping region. The higher doping of the n-doped region results in the lower resistance, which requires more power for the same heating field. However, if the doping concentration is too low, the carrier population will not be enough. Our aim is to raise the doping to $5 \times 10^{16} \text{ cm}^{-3}$. The simulation results, Figure 10, show that there is no difference for hole concentrations in both quantum well and the n-doped region when the doping of the n-doped region is changed to $5 \times 10^{16} \text{ cm}^{-3}$.

One may notice a slightly higher, but still less than 10^{12} cm^{-3} , of electron concentration in the quantum well. This higher electron concentration is basically from the higher doping of the n-doped region but as it is very low, it shows that the barrier is working

perfectly. Therefore we can safely change the doping concentration of the n-doped region to $5 \times 10^{16} \text{ cm}^{-3}$.

So far, it seems that all parameters have been obtained. However, the middle part of the structure is the p⁻-doped region, which is doped at $1 \times 10^{15} \text{ cm}^{-3}$. This doping level might not be easy to achieve when growing the wafer. Therefore, we decided to leave that region undoped. When the $\text{Al}_x\text{Ga}_{1-x}\text{As}$ is undoped, the built-in doping level is $1 \times 10^{15} \text{ cm}^{-3}$ n-type. Another simulation was performed to see if the change of the doping would affect any results we previously have. Figure 11 shows the concentration of carriers at an applied voltage of 2V and indicates that the results are insensitive to the doping in the central region where the range of doping is from p-doped of $1 \times 10^{15} \text{ cm}^{-3}$ to n-doped of $1 \times 10^{15} \text{ cm}^{-3}$. This is reasonable as the doping is very low and the middle region is almost totally depleted. We also found that all results at every bias are almost identical.

4. Conclusion

From the results, the appropriate structure was found as shown in Figure 12. However, according to the result shown in Figure 9, the material of the n-doped region will be $\text{Al}_{0.1}\text{Ga}_{0.9}\text{As}$. As mentioned earlier, the mobility of $\text{Al}_{0.1}\text{Ga}_{0.9}\text{As}$ is lower than GaAs, which makes the device more difficult to heat. The suggestion is that there should be two testing wafers with different materials in the n-doped region, i.e. $\text{Al}_{0.1}\text{Ga}_{0.9}\text{As}$ and GaAs.

Acknowledgements

The first author would like to specially thank to Thailand Research Fund (TRF) for funding and supporting. Special thanks also go to the Department of Physics, Faculty of Science, Prince of Songkla University, Thailand for allowing the first author to work on this project.

References

- [1] M. S. Borella, J. P. Jue, D. Banerjee, B. Ramamurthy, and B. Mukherjee, "Optical Components for WDM Lightwave Networks", *Proceedings of the IEEE*. **85**, 1274 (1997).
- [2] F. Sugihwo, M. C. Larson, and J. S. Harris, Jr., "Low threshold continuously tunable vertical surface-emitting lasers with 19.1 nm wavelength range", *Applied Physics Letters* **70**, 547 (1997).
- [3] P. Wang, P. Tayebati, D. Vakhshoori, C. Lu, "Half-symmetric cavity microelectromechanically tunable vertical cavity surface emitting lasers with single spatial mode operating near 950 nm", *Applied Physics Letters* **75**, 897 (1999).
- [4] F. E. Reed, D. Zhang, T. Zhang, and R. M. Kolbas, "Three-terminal bias induced dual wavelength semiconductor light emitter", *Applied Physics Letters* **65**, 570 (1994).
- [5] M. Kuznetsov, "Design of Widely Tunable Semiconductor Three-Branch Lasers", *Journal of Lightwave Technology* **12**, 2100 (1994).
- [6] H. Ishii, H. Tanobe, F. Kano, Y. Tohmori, Y. Kondo, and Y. Yoshikuni, "Broad-range wavelength coverage (62.4nm) with superstructure-grating DBR laser", *Electronics Letters* **32**, 454 (1996).
- [7] R. C. Alferness, U. Koren, L. L. Buhl, B. I. Miller, M. G. Young, T. L. Koch, G. Raybon, and C. A. Burrus, "Broadly tunable InGaAsP/InP laser based on a

vertical coupler filter with 57-nm tuning range", *Applied Physics Letters* **60**, 3209 (1992).

[8] V. Comincioli, *Numeric Analysis*, McGraw Hill, New York, 1990.

[9] H. Schildt, *C++ from the Ground Up*, 2nd edition, Osborne McGraw-Hill, Berkeley, USA, 1998.

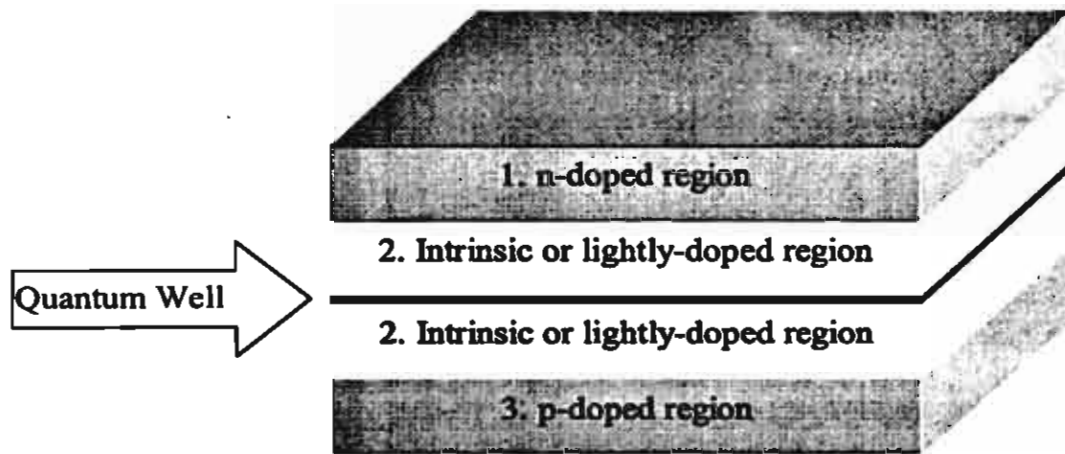


Figure 1. The outline of the device structure whose doping concentration, percentage of aluminium in $\text{Al}_x\text{Ga}_{1-x}\text{As}$ of all layers, size and position of the quantum well were altered and simulated.

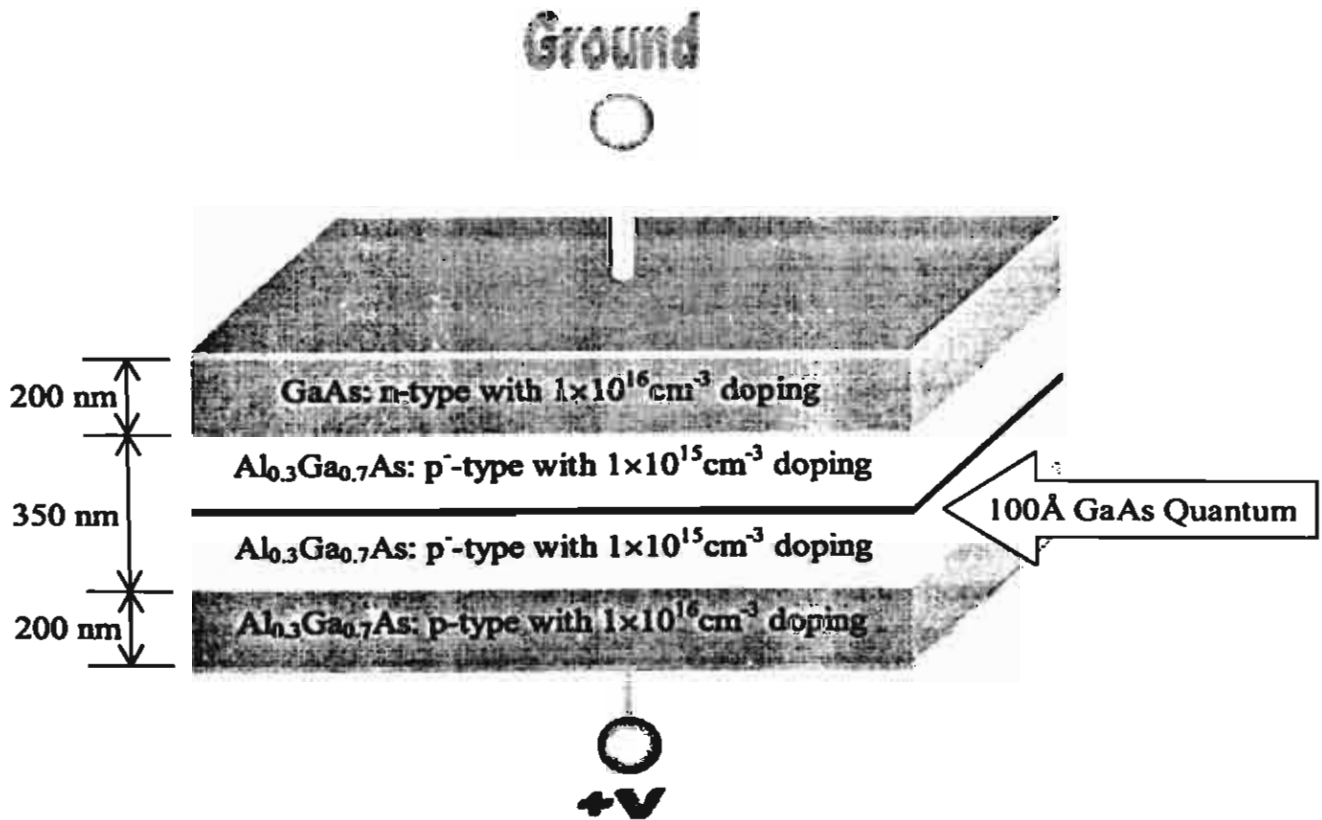


Figure 2. The first structure for simulation is shown with the dimensions and doping concentrations. The quantum well is in the middle of the p-type region.

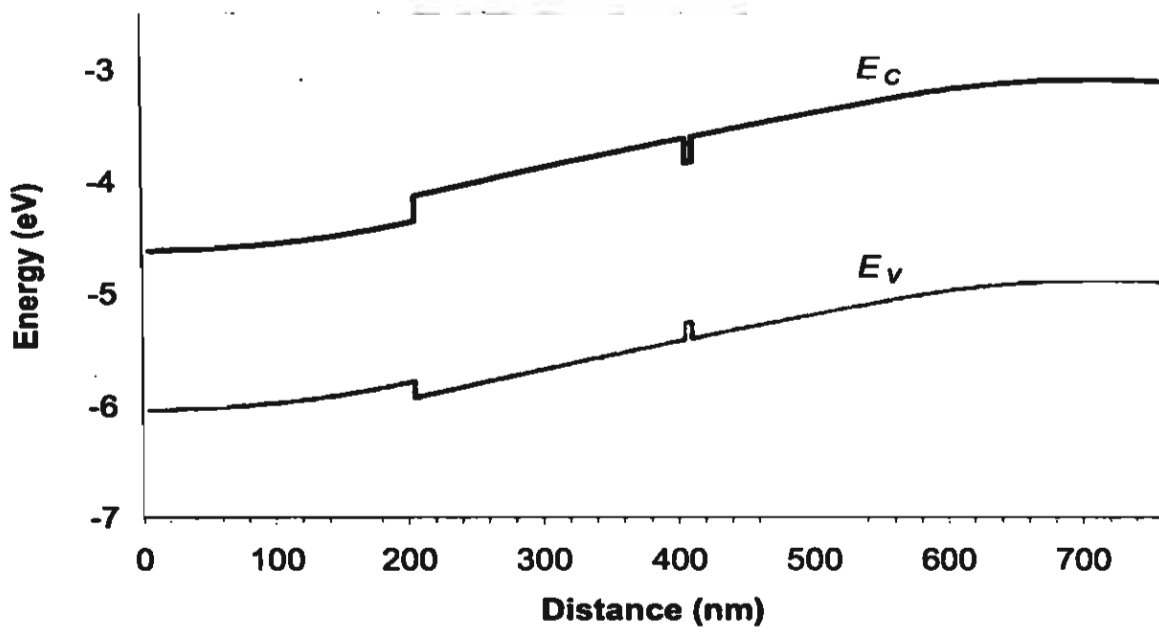


Figure 3. The band edge diagram of the first device in equilibrium.

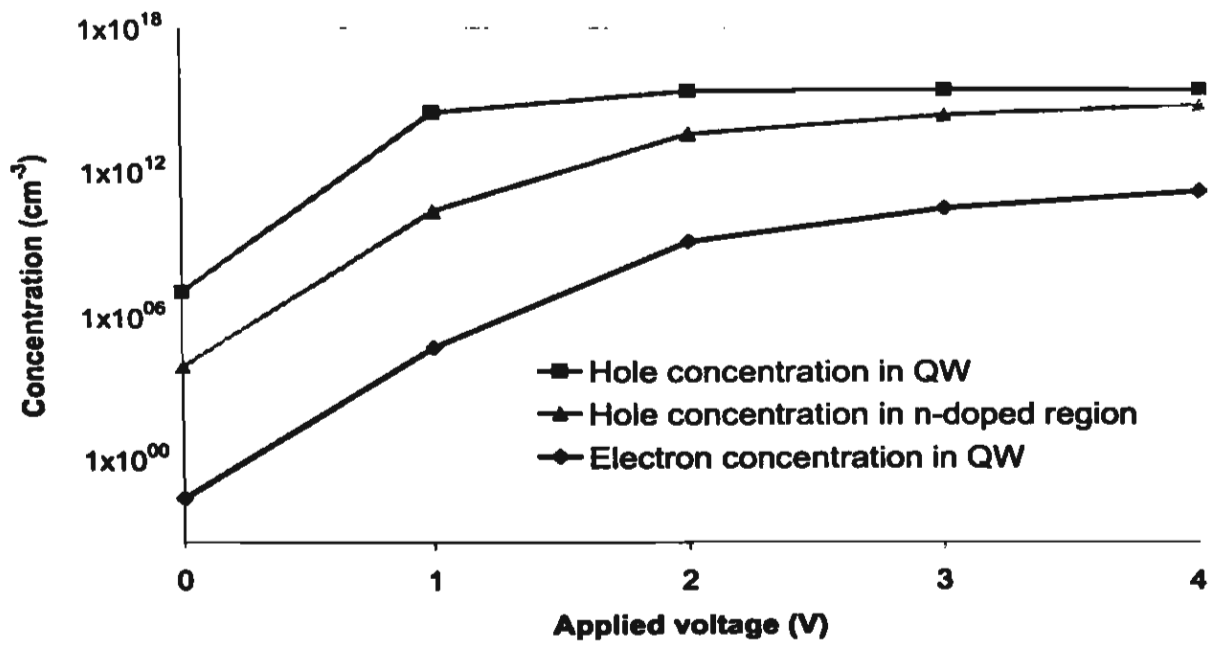


Figure 4. Carrier concentrations with the different applied voltages of the first device.

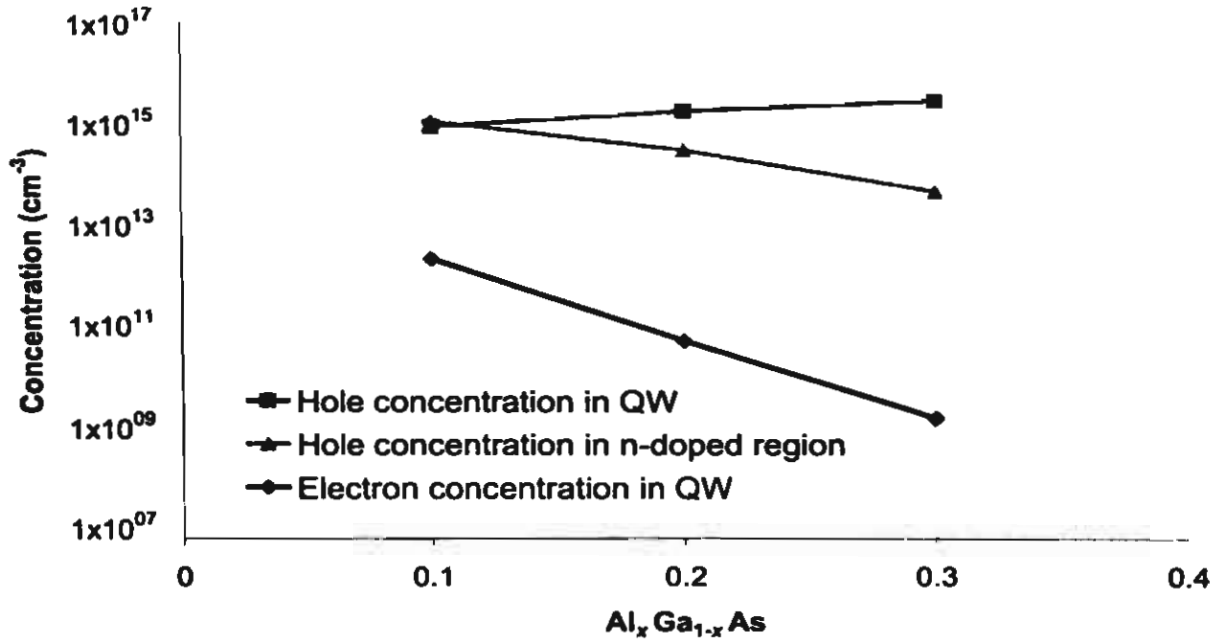


Figure 5. Concentrations in the quantum well and n-doped region of the devices with different Al composition in the p⁻ and p-doped regions. The devices are two-volt forward biased.

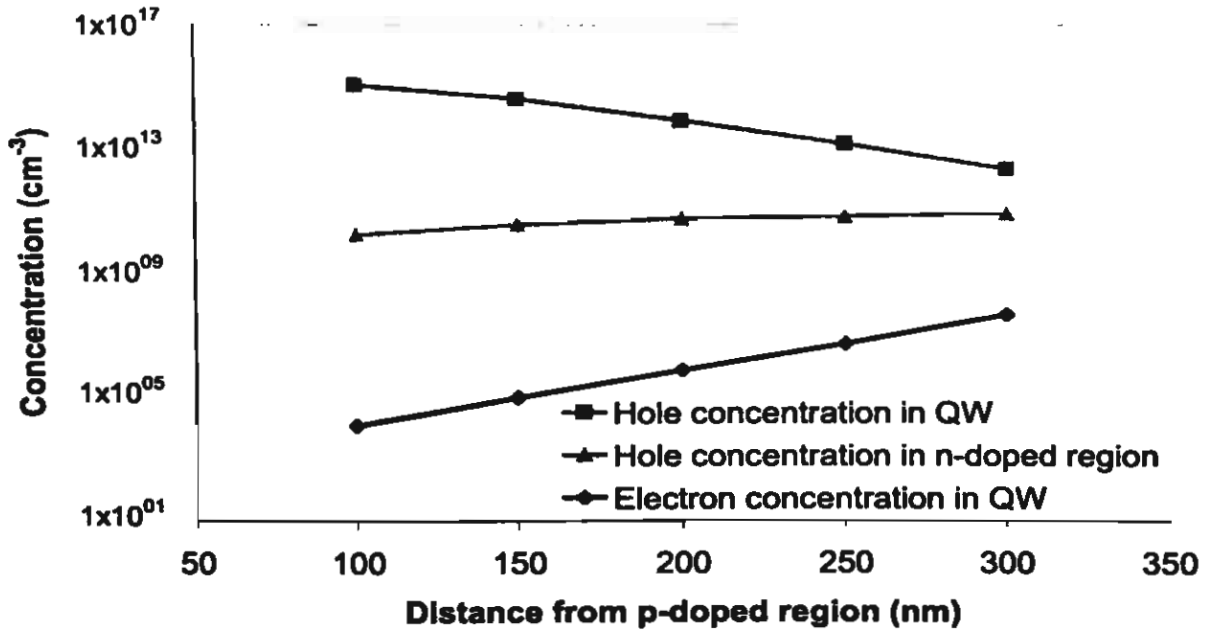


Figure 6. Concentrations in the quantum well and n-doped region of the devices with different position of the quantum well. The devices are one-volt forward biased.

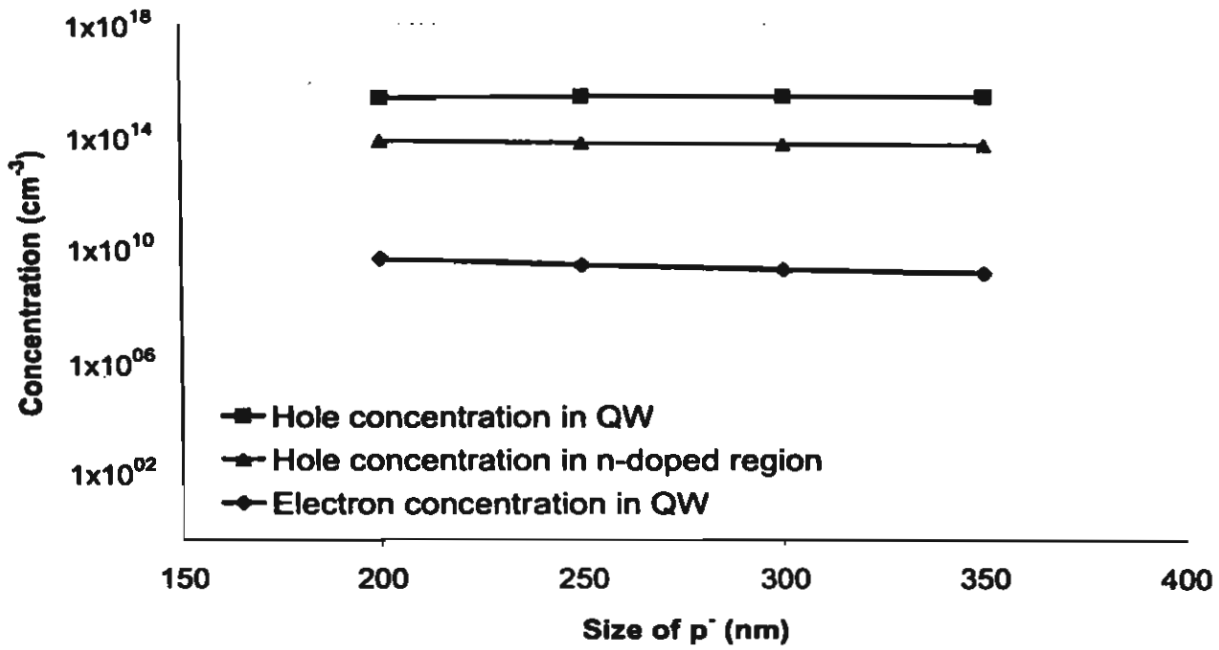


Figure 7. Concentrations in the quantum well and n-doped region of the devices with different sizes of the p--doped region. The devices are two-volt forward biased.

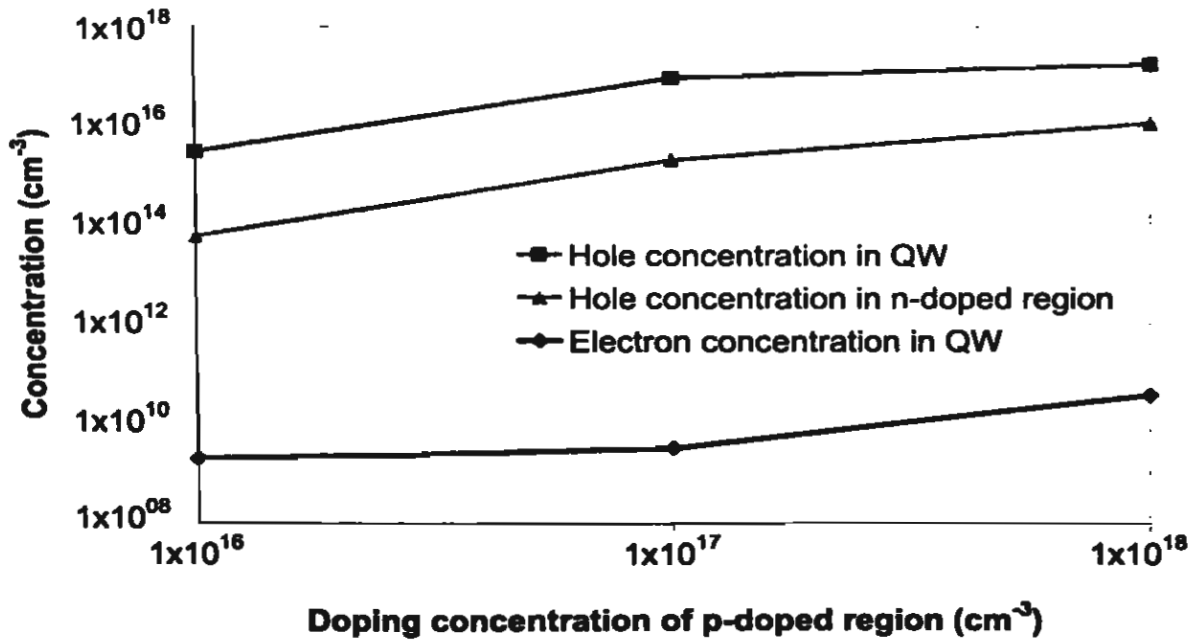


Figure 8. Concentrations in the quantum well and n-doped region of the devices with different doping of the p-doped region. The devices are two-volt forward biased.

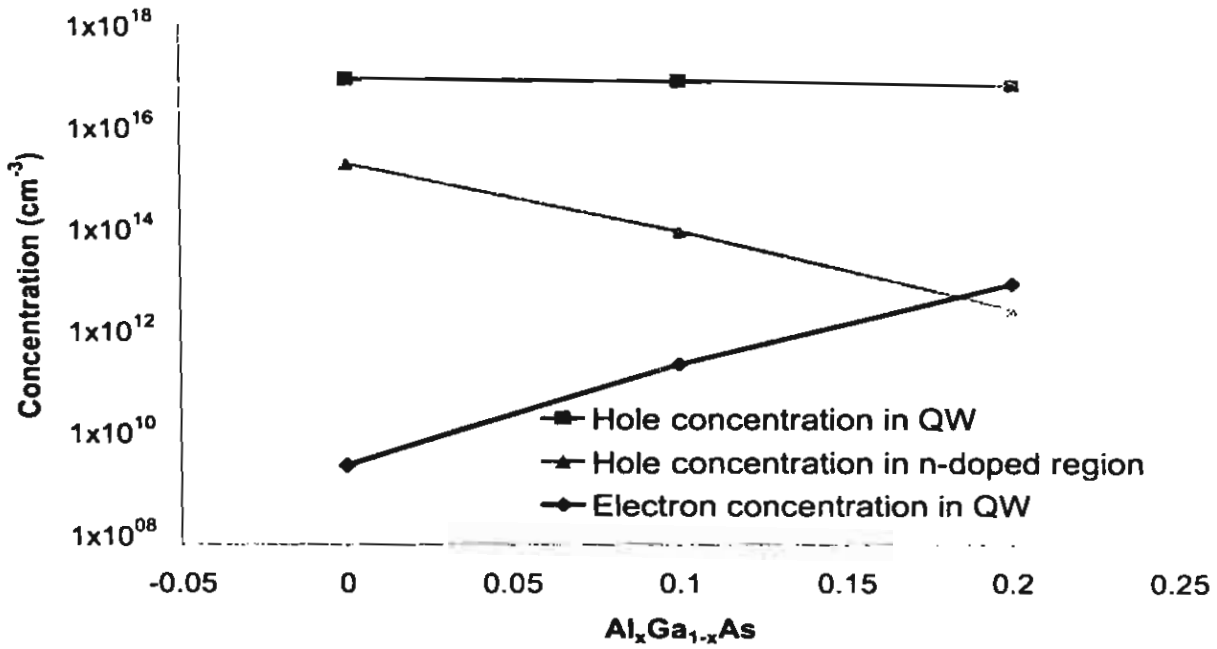


Figure 9. Concentrations in the quantum well and n-doped region of the devices with different Al composition in n-doped region. The devices are two-volt forward biased.

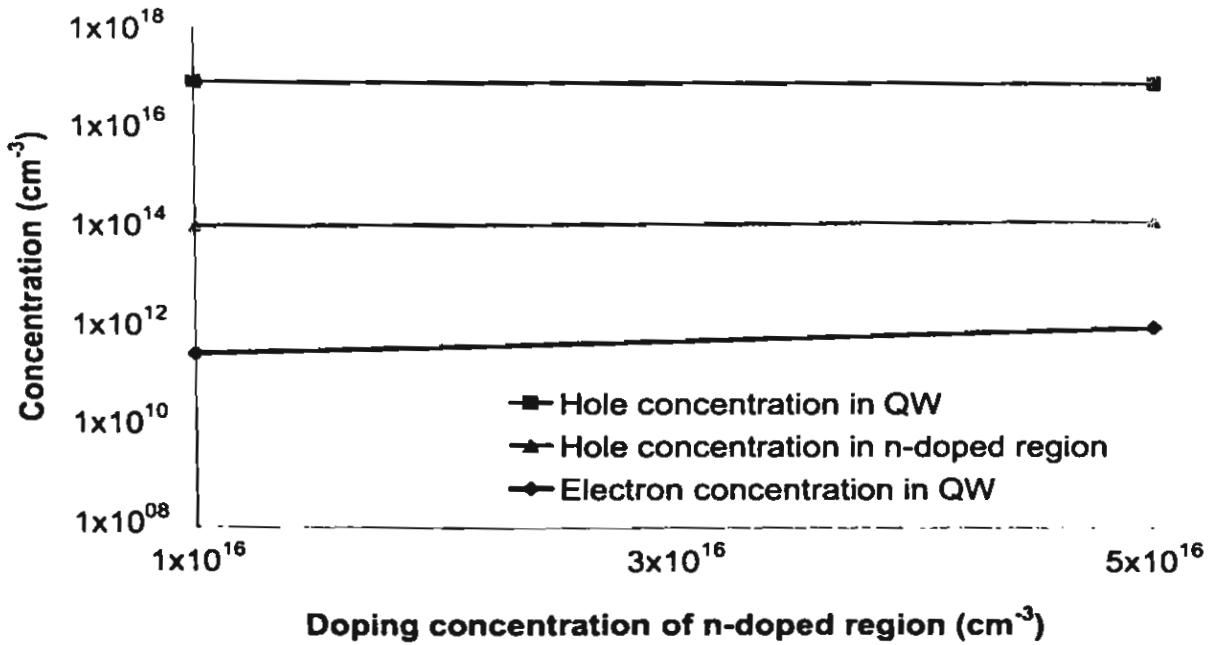


Figure 10. Concentrations in the quantum well and n-doped region of the devices with different doping of the n-doped region. The devices are two-volt forward biased.

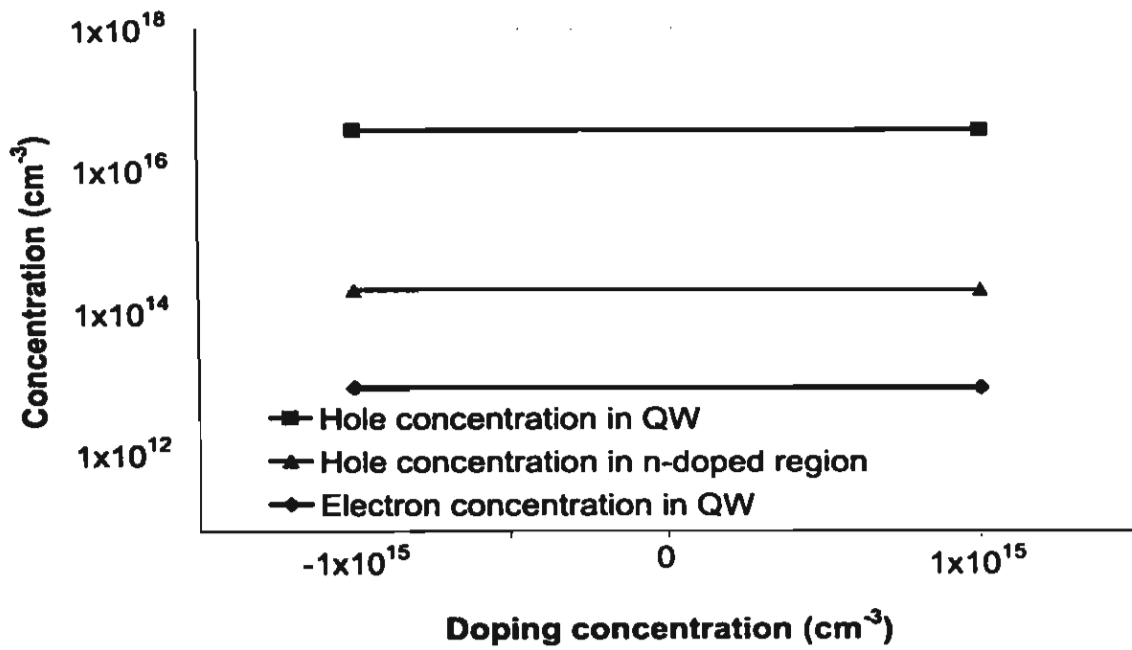


Figure 11. Concentrations in the quantum well and n-doped region of the devices with different doping of the middle region, where the negative doping means p-typed doping and the positive means n-typed doping. The devices are two-volt forward biased.

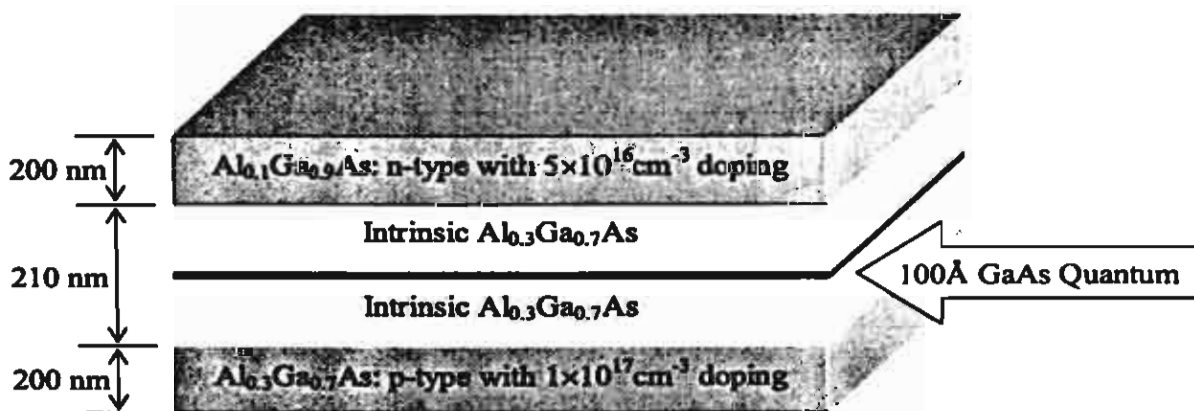


Figure 12. The optimized structure for the device. The quantum well is in the middle of the intrinsic region.